Development of an adaptive multi-resolution method to study the near wall behavior of two-dimensional vortical flows

Seyed Amin Ghaffari, Kai Schneider

Laboratoire de M2P2-UMR 7340, CNRS et Les Universités d'Aix-Marseille
Directeur de labo: Patrick Bontoux (bontoux@L3m.univ-mrs.fr)

Rapport de recherche :

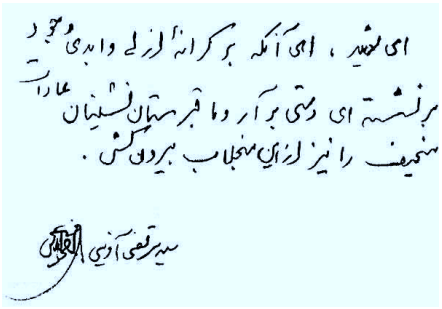
Développement d'une méthode multirésolution adaptative pour étudier le comportement des écoulements tourbillonnaires près des parois solides

***

Development of an adaptive multiresolution method to study the near wall behavior of two-dimensional vortical flows

Author: Seyed Amin Ghaffari (ghaffari@L3m.univ-mrs.fr)
Co author: Kai Schneider (kschneid@cmi.univ-mrs.fr)

March 2014



Oh who is sitting at the eternal extreme of the existence,
help me to rescue from the cemetery of my bad habits.

Morteza Avini

Développement d'une méthode multirésolution adaptative pour étudier le comportement des écoulements tourbillonnaires près des parois solides

Résumé

L'objectif de cette recherche est de développer[1] une méthode multi-échelle adaptative en temps et en espace plus efficace que les méthodes actuelles pour résoudre des équations aux dérivées partielles de type hyperbolique intervenant en mécanique des fluides. La nouvelle méthode, basée sur une discrétisation en différences finies d'ordre deux et une analyse multi-échelle, permet de réduire significativement le nombre de points nécessaires. La grille se raffine automatiquement dans les régions avec un fort gradient. La méthode est appliquée à l'équation de Burgers puis prolongé aux équations de Navier-Stokes bidimensionnelles. Pour étudier le comportement des écoulements tourbillonnaires près des parois solides, le problème du collision de dipôle sur paroi droit a été utilisé comme référence et ensuite le collision sur des parois courbes avec la méthode de pénalisation a été considéré. La stratégie d'adaptation est basée sur la transformée en ondelettes et le seuillage des coefficients. Pour intégrer les équations dans le temps, on utilise des méthodes de Runge-Kutta d'ordre différents, avec un pas de temps fixe ou adaptatif. Les résultats obtenus montrent que le temps de calcul peux être réduit considérablement avec cette méthode en conservant la précision.

Mots clés: Ecoulements incompressibles, Analyses multirésolutions , Formulation fonction de courant et vorticité, Méthode de pénalisation en volume, Différence fini centrée

[1] Le cose est développée en FORTRAN et accessible à tous sur demande par mail [55].

Development of an adaptive multiresolution method to study the near wall behavior of two-dimensional vortical flows

Abstract

In the present investigation, a space-time adaptive multiresolution method is developed[1] to solve evolutionary PDEs, typically encountered in fluid mechanics. The new method is based on a multiresolution analysis which allows to reduce the number of active grid points significantly by refining the grid automatically in regions of steep gradients, while in regions where the solution is smooth coarse grids are used. The method is applied to the one-dimensional Burgers equation as a classical example of nonlinear advection-diffusion problems and then extended to the incompressible two-dimensional Navier-Stokes equations. To study the near wall behavior of two-dimensional vortical flows a recently revived, dipole collision with a straight wall is considered as a benchmark. After that an extension to interactions with curved walls of concave or convex shape is done using the volume penalization method. The space discretization is based on a second order central finite difference method with symmetric stencil over an adaptive grid. The grid adaptation strategy exploits the local regularity of the solution estimated via the wavelet coefficients at a given time step. Nonlinear thresholding of the wavelet coefficients in a one-to-one correspondence with the grid allows to reduce the number of grid points significantly. Then the grid for the next time step is extended by adding a safety zone in wavelet coefficient space around the retained coefficients in space and scale. With the use of Harten's point value multiresolution framework, general boundary conditions can be applied to the equations. For time integration explicit Runge-Kutta methods of different order are implemented, either with fixed or adaptive time stepping. The obtained results show that the CPU time of the adaptive simulations can be significantly reduced with respect to simulations on a regular grid. Nevertheless the accuracy order of the underlying numerical scheme is preserved.

Keywords: Incompressible flows, Multiresolution analysis, Vorticity stream-function formulation, Volume penalization method, Central finite differences

[1] The code is developed in FORTRAN and is accessible for all [55].

Contents

Chapter 1

Introduction

1.1 Objectives and historical background

The aim of present investigation is to develop a reliable self-adaptive direct numerical method to study the near wall behavior of unsteady incompressible flows. The governing equations are the Navier-Stokes equations with proper initial and boundary conditions. Conventional methods for spatial discretization of the PDEs (e.g., finite differences, finite volumes and finite elements) have limited order of accuracy especially near boundaries, but they are more flexible in dealing with complex geometries over a suitable grid. On the other hand standard spectral methods which are widespreadly used in direct numerical simulation of turbulence are limited to Cartesian grids. One can recognize the poor spectral localization (good spatial localization/resolution) of the former methods while good spectral localization (poor spatial localization/resolution) of the latter methods [25]. Therefor for flows in/around complex geometries, the use of a grid capable to resolve all the scales will be limited to low-Reynolds flows. In general the small scales in a turbulent flow are not limited to near wall regions and can also move in the flow field. A high-order method with a prescribed grid spacing cannot resolve all the scales and thus the interactions present in the flow unless with the use of a fine grid in regions with high gradient. The limitation of mentioned methods for problems with widely disparate spatial scales, has encouraged the researcher to use alternative methods with limited accuracy but good spatial localization in regions with high gradient of flow variables. As a result recently there has been increasing interest in self adaptive numerical methods for solving time-dependent PDEs, see for example [46]. Adaptive methods can be divide into r-type (a fixed number of grid points are redistributed), h-type (regriding is performed occasionally) and p-type (the degree of the polynomial representing the solution is locally increased) each with their own advantages and disadvantages as detailed in literatures. Among different methods for grid adaptation h-type refinement proved to be more advantageous. By considering some advantages of the h-type refinement we explain some details of this methods. Existing numerical methods based on h-type refinement fall into two classes: error indicator based, where the grid is refined to resolve gradients of a physically relevant quantity, and error control based, where the

error is estimated and the grid is refined to ensure this error is less than a prescribed tolerance. The error-indicating strategy does not control the error directly, but instead controls the mesh coarsening and refinement. The error-estimating strategy minimizes the error as measured in an appropriate norm, which leads to an optimal distribution of the grid and thus a more reliable solution.
Among different error-estimating adaptation strategies (which most of them belong to the finite element family) wavelet-based numerical methods have proved to be an efficient tool in developing adaptive numerical methods which control the global (usually L_2) approximation error. Wavelet transforms allow to estimate the local regularity of solutions of the given PDE, with a very efficient algorithm, and thus can define auto-adaptive discretization with local mesh refinement. Liandrat and Tchamitchian [11] proposed the first wavelet-based adaptive method for numerical simulation of PDEs. The currently existing wavelet-based algorithms can be classified as pure wavelet methods and wavelet optimized grid methods. Pure wavelet methods, divided to Galerkin and collocation schemes, employ wavelets directly for discretization of the governing equations. The wavelet transform facilitates the effective sparse representation of the functions and pseudo-differential operators (and their inverse) by filtering of wavelet coefficients of the function and of the matrix representing the operators. The characterization of function spaces in terms of wavelet coefficients and the corresponding norm equivalences leads to diagonal preconditioning of operators (compression) in wavelet space. In practice similar to pseudo-spectral methods the evaluation of nonlinear term in function space is not efficient, usually by performing an inverse wavelet transform in each time step, the solution is reconstructed on a locally refined grid, and the nonlinear term is then evaluated point-wise in physical space. For a comprehensive review of wavelet methods applied to the computational fluid dynamics see [52]. On the other hand, wavelet optimized grid (WOG) methods combine classical discretizations of considered equations (e.g., finite differences or finite volumes) with wavelets, which are used to define the adaptive grid. The goal of the wavelet-based adaptation of the grid is to obtain the best approximation of the flow variables on a near optimal grid. This approximation has a one-to-one correspondence between the wavelet expansion coefficients and the grid points. Thus, nonlinear filtering of wavelet coefficients of the interested flow variables automatically coarsens the computational grid. Since with the use of wavelet transform and thresholding of the wavelet coefficients, a given functions can be reconstructed with a prescribed accuracy, thus adaptive methods based on wavelets provide global error control for numerical solution of PDEs. See [33] and [53] where a finite volume discretization of governing equations combined with cell-averaged interpolating wavelet transform for grid adaptation. In the present work the Navier-Stokes equations are discretized with a classical second order central finite differences, thus similar to WOG methods the role of the wavelet transform is the adaptation of the grid and the fast interpolation of flow variables at new unknown points. Wavelet optimized finite difference (WOFD) methods based on fast wavelet transforms are very efficient (especially for problems with a large magnitude of scales, e.g., turbulence) since the computational cost scales linearly with the number of wavelet coefficients retained in the approximation [43].
In the procedure of solving the incompressible Navier-Stokes equations a difficulty

arises because of the lack of a time-derivative term in the continuity equation, which limits the straightforward use of time-marching numerical methods designed for coupled equations. Another problem is the procedure of updating the pressure, which requires special treatment. Most numerical methods like projection, marker and cell and fractional step methods involve solving a decoupled Poisson equation for the pressure at each time step. The common application of this method consist of two steps: The first step is to solve for an intermediate velocity field using the momentum equations (Burgers-like), in which a fraction of the pressure-gradient term from the previous time step can be involved or it can be excluded entirely. In the second step, the pseudo pressure is computed via intermediate velocity field, which maps (Leray projector) the intermediate velocities onto a divergence-free (selonoidal) velocity field. This method is first order in time, originally proposed by Chorin [3] in 1968 and developed by Kim and Moin [9] to second order accuracy. For a comprehensive review of projection methods see [40]. Another approach is the artificial compressibility method (ACM), which relaxes the ellipticity of the pressure field by adding a pseudo-time derivative of pressure into the continuity equation, hence coupling continuity and momentum equations. Then iterations are performed in pseudo-time until attaining a divergence free velocity field. The method was originally designed for steady problems by Chorin [2] in 1967 and developed by Rogers and Kwak [13] to second order accuracy in time, with the use of dual time stepping strategy. For a comprehensive review of ACM methods see [36]. However in two-dimensional problems vorticity and stream-function formulation would be the best for some purposes by eliminating the pressure gradient entirely and reducing the flow variables to one scalar value.

In the present work the concept of adaptive multiresolution method will be applied to the vorticity and stream-function formulation. However the developed concepts are also applicable to the primitive variable formulation. See [32] and [39] for a finite difference discretization of governing equations using gradient approach for grid adaptation. To the best of our knowledge this investigation is the first attempt to develop an adaptive finite difference method based on the interpolating wavelet transform applied to the vorticity and stream-function formulation. A second-order central finite difference method with symmetric stencil over an adaptive Cartesian grid is used for spatial discretization of the equations. Finite difference method represents a suitable combination with the multiresolution analysis based on the Harten's point-value wavelet transform. The chosen discretization avoids adding excessive numerical dissipation which is usually introduced by upwind methods, thus has a reasonable accuracy in space over an adapted grid. The concept of symmetric stencils will lead to intermediate (hung) points, that their values can be interpolated accurately via inverse wavelet transform. Another advantage of the vorticity and stream-function formulation is that it allows the use of high-order time stepping rather easily, e.g., fourth-order Runge-Kutta method, to minimize the truncation error due to discrete time integration which is in nature an accumulative error during simulations [28].

In the present investigation the unexpectedly difficult, dipole-wall collision is chosen as a benchmark computation. The problem is studied experimentally with the use of PIV technique by Naguib and Koochesfahani [35]. Numerical simulations of the dipole-wall collision were first performed by Orlandi [12] in two dimensions, who

revealed that the creation of vorticity at the wall drives a vigorous rebound of the dipolar vortex. Each dipole half forms a new pair with a secondary vortex consisting of boundary layer vorticity. The new dipoles travel along circular trajectories away from the wall. The production of the secondary vorticity at the wall during the dipole-wall collision was later studied by Coutsias and Lynov [14] and by Clercx and van Heijst [29]. The Reynolds number in the reported simulations is limited to $Re = 5000$ as accurate representation of the small-scale structures near the wall requires very high resolution. Walker [5] and Peridier et al. [15, 16] have investigated the behavior of the boundary layer at a no-slip wall affected by a nearby single (point) vortex. This setup permits the use of boundary-layer equations, i.e., a reduced version of the Navier-Stokes equations, on a specially tailored grid to obtain results for large Reynolds numbers. More recently Obabko and Cassel [30] reported results on this problem based on the full Navier-Stokes equations [45].
After validation of the developed adaptive multiresolution method with the results of previous studies of a dipole collision with a straight wall the extension to collision with curved walls via the volume penalization method will be presented. Nowadays, there exist several techniques to introduce solid curved boundaries in the numerical simulation of different types of flows. Among those, some of them use finite difference methods by mapping the equations to general curvilinear coordinates [13], others use full unstructured grids with finite volume or element discretization of the governing equations, we will call them body-fitted grids. All of them have their own advantages and difficulties. Another approach which became more popular during recent years are immersed boundary methods for imposing complex geometries while using the advantages of Cartesian grids. They are easy-to-implement and more efficient than classical approaches such as body-fitted grids, in particular for moving and/or flexible geometries [44]. The name *immersed boundary* comes from the fact that the physical solid boundary does not always conform with the computational grid points at boundaries, instead it is inside of the solution domain. The volume penalization method originally introduced by Arquis and Caltagirone [7] for flows in porous media, does posses this property; therefore it is an example of immersed boundary methods. This technique is physically motivated, since it is based on the idea of modeling solids as porous media whose porosity coefficient tends to zero. In addition, it is mathematically justified, due to the convergence property proven rigorously by Angot et al. [23].
As a starting point in the present work we take the two-dimensional vorticity streamfunction solver developed in [49] for a uniform grid and the adaptive one-dimensional Burgers solver developed by author at M2P2 (Marseilles) during the last summer.
The manuscript is organized as follows: In the following a short introduction to the discrete wavelet transforms and the idea of point selection by filtering of the wavelet coefficients will be presented. Then in chapter two some theoretical background of the adaptive multiresolution method will be introduced. After that the results for the one-dimensional Burgers equation and the dipole collision with either straight or curved walls will be reported. Finally, conclusions and perspectives will be discussed in chapter four.

1.2 Wavelet transform

Like Fourier transforms, wavelet transforms can be viewed as a change of basis in function space, e.g., from time to frequency or from space to wavenumber. The basis functions of the Fourier transform are trigonometric functions, i.e., sines and cosines. For the wavelet transform there is a large choice of basis functions which are called mother wavelets. For the Fourier transform the decay of the Fourier coefficients depends on the global regularity of the decomposed function. In wavelet transforms some of the new basis functions are more complicated than trigonometric functions but what makes them interesting is that, unlike the Fourier basis, wavelets are well localized functions in space; simultaneously, like them, they are quit localized in function space or more precisely they have a characteristic scale. So they can unfold signals or fields into both space (or time) and scale, and possibly directions in dimensions higher than one. The continuous wavelet transform has been discovered by Grossmann and Morlet who published the first paper on wavelets in 1984 [8]. The orthogonal wavelet transform has been discovered by Lemarié and Meyer (1986). Then, Daubechies (1988) found orthogonal bases made of compactly supported wavelets, and Mallat (1989) designed the fast wavelet transform (FWT) algorithm. More details can be found in [41]. The second generation wavelet transform (SGWT) was introduced independently by Sweldens [22], Donoho [17] and Harten [20] in the early 1990s. It is a wavelet transform where the filters (or even the represented wavelets) are not designed explicitly, but rather the transform consists of calculating the differences between an interpolated value (with different accuracies) of the function at a given point from the corresponding neighbor points in lower levels, and its value. The main advantage of this approach is that it is applicable to either periodic or non-periodic functions.

Biorthogonal wavelet transform

In a discrete multiresolution analysis framework, data are represented at different scale levels, and the main tools are appropriate transformations, e.g., wavelet transform, relating the information f_J at the finest scale level J to the lower ones, and vice versa. After a transformation the output contains the information f_0 on the coarsest level, and d_j that keeps the details between a scale level j and the next upper level $j+1$. Schematically, we have

$$f_J \leftrightarrow f_J^{MR} = (f_0, d_0, ..., d_{J-1}) \tag{1.1}$$

To illustrate this methodology, we consider the case of discretization by Harten's point values [21] for uniform grids, which is well adapted for finite difference methods, versus Harten's cell average method which is more suitable for finite volume methods. By considering in a unit interval the hierarchy of uniform dyadic grids will obtain from

$$X_j = \{x_{j,i} \in \mathbb{R} : x_{j,i} = i2^{-j}, i = 0, \cdots, 2^j\},\ j = 0, \cdots, J \tag{1.2}$$

with spacing 2^{-j}, where j is the level and i represents the position. The number of points must always be odd ($N = 2^J + 1$) to have a point in the middle. As indicated in Fig. 1.1, in multiresolution analysis to go from X_j to a more refined grid X_{j+1},

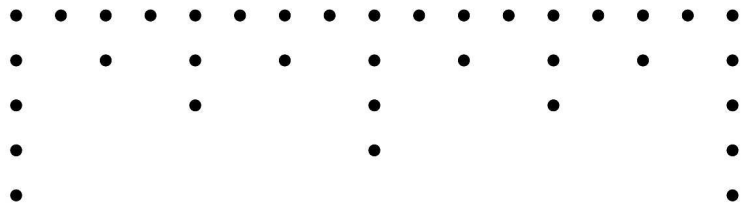


Figure 1.1: Uniform dyadic grids with $N = 2^J + 1 = 17$ points in finest level ($J = 4$), including lower levels down to ($j = 0$).

we add to X_j new midpoints between the old points, dividing by two the step size. The direct and inverse multiresolution (MR) transforms are usually associated to bi-orthogonal multiresolution representation of functional spaces of the form

$$V_J = V_0 \oplus \sum_{j=0}^{J} W_j \tag{1.3}$$

where V_J and W_J are associated function spaces. Thus a given function, $f(x)$ can be represented with

$$f(x) = \sum_{i=0}^{2^J} f_{0,i}\Phi_{0,i}(x) + \sum_{j=0}^{J}\sum_{i=0}^{2^J} d_{j,i}\Psi_{j,i}(x) \tag{1.4}$$

where the orthonormal basis are scaling functions $\Phi_{j,i}$ and wavelets $\Psi_{j,i}$ defined as

$$\Psi_{j,i}(x) = 2^{j/2}\Psi(2^j x - i), \quad j, i \in \mathbb{Z}$$

$$\Phi_{j,i}(x) = 2^{j/2}\Phi(2^j x - i), \quad j, i \in \mathbb{Z}$$

At a given scale (j) the scaling function $\Phi_{j,i}$ is orthonormal with respect to its translation by discrete steps $i2^{-j}$ but not with respect to its dilates,

$$\langle \Phi_{j,i}, \Phi_{j,k} \rangle = \delta_{ik} \tag{1.5}$$

The wavelets $\Psi_{j,i}$ are orthonormal with respect to their translates by discrete steps $i2^{-j}$ and their dilates by discrete steps 2^{-j} corresponding to octaves [41].

$$\langle \Psi_{j,i}, \Psi_{j',i'} \rangle = \delta_{jj'}\delta_{ii'} \tag{1.6}$$

where δ denotes the Kronecker symbol and the inner (scalar) product is defined by

$$\langle f, g \rangle = \int_{\mathbb{R}} f(x)g^*(x)dx \tag{1.7}$$

where $*$ denotes the complex conjugate in the case of complexed valued wavelets. The scaling function coefficients, $f_{j,i}$, essentially encode the smooth part of the function, while the wavelet coefficients, $d_{j,i}$, contain information on the function behavior on successively finer scales and the coefficients are interpreted as the inner products with the dual basis

$$f_{j,i} = \langle f, \Phi_{j,i} \rangle = f(x_{j,i}) \tag{1.8}$$

where $\Phi_{j,i}$ is the delta distribution at $x_{j,i}$. Definition of the dual wavelets $\Psi_{j,i}$ such that

$$d_{j,i} = \langle f, \Psi_{j,i} \rangle = f_{j+1,2i+1} - \tilde{f}_{j+1,2i+1} \tag{1.9}$$

implies

$$\Psi_{j,i}(x) = \Phi_{j+1,2i+1} - \sum_n \beta_n \Phi_{j,n}(x) \tag{1.10}$$

thus $\tilde{f}$, can summarized in

$$\tilde{f}_{j+1,2i+1} = \sum_{n=1}^{st/2} \beta_n (f_{j,i-n+1} + f_{j,i+n}) \tag{1.11}$$

where st is the interpolation stencil. For instance, interpolating coefficients for linear interpolation with $st = 2$ is $\beta_1 = 1/2$, and for cubic interpolation with $st = 4$ are $(\beta_1, \beta_2) = (9/16, -1/16)$. The following equation is useful to better understand the relation of indexes between two successive level for a given variable ϕ

$$\phi_{j+1,2i+1} \Longleftrightarrow \phi_{j,i+\frac{1}{2}} \tag{1.12}$$

1.2.1 One-dimensional transform

The direct wavelet transform (WT) for a discrete function given on dyadic grid points, is performed simply by replacing the value of the function, with the difference between its value and its interpolated value from the corresponding neighbors at lower levels

$$d_{j,i} = f_{j+1,2i+1} - \tilde{f}_{j+1,2i+1} \tag{1.13}$$

this must be performed from the finest level J down to $j = 1$. In contrast the inverse wavelet transform (IWT) starts from the coarsest (lowest) level up to the finest level

$$f_{j+1,2i+1} = d_{j,i} + \tilde{f}_{j+1,2i+1} \tag{1.14}$$

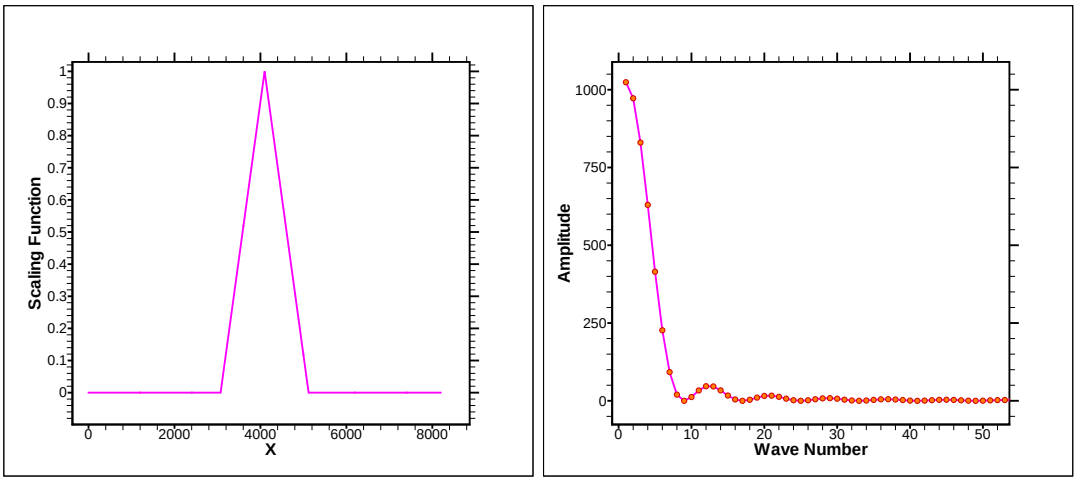
In this method predictions of the values of the function will be done with interpolating polynomials. The prediction is exact for polynomials of a prescribed degree [54]. The most simple example is given by linear (first-order) interpolation

$$\tilde{f}_{j+1,2i+1} = \frac{f_{j,i} + f_{j,i+1}}{2} \tag{1.15}$$

By using cubic (third-order) interpolation

$$\tilde{f}_{j+1,2i+1} = \frac{-f_{j,i-1} + 9f_{j,i} + 9f_{j,i+1} - f_{j,i+2}}{16} \tag{1.16}$$

better results will obtained in terms of compression rate for sufficient smooth functions. The corresponding scaling functions of the wavelet transform in physical and Fourier space are shown in Fig. 1.2 and Fig. 1.3 with linear and cubic interpolation, respectively. The functions can be obtained by computing the inverse wavelet



(a) In physical space (b) Modulus of its Fourier transform

Figure 1.2: The scaling function $\Phi_{1,i}$ of wavelet transform corresponding to linear interpolation.

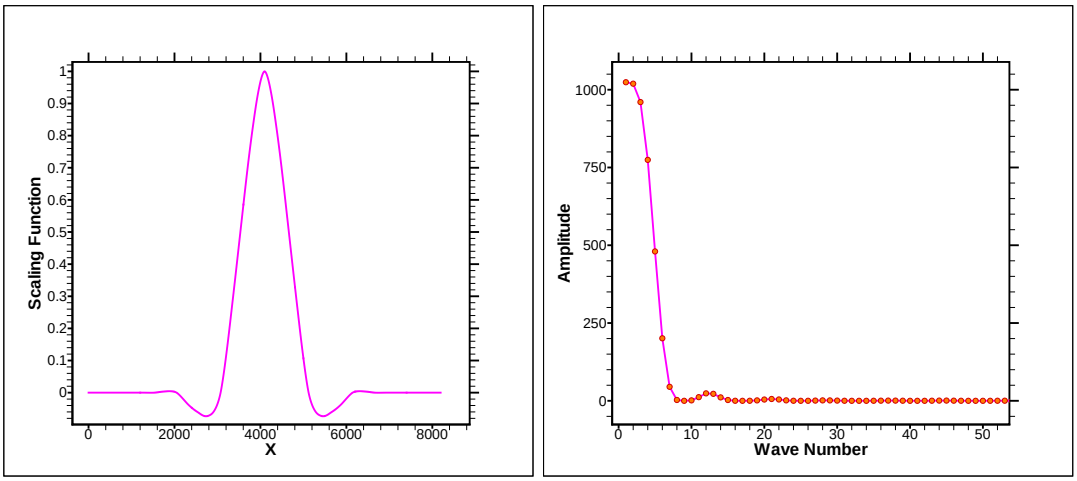
transform of a unit vector (Dirac-delta function δ) in the mid component of a zero vector (of length, e.g., 8192) so the point is belonging to the minimum level. Other unit vectors would give wavelets with the same shapes, but different positions and scales. The smooth scaling functions, e.g., the one obtained with cubic interpolation (see Fig. 1.3) decays rapidly in Fourier space. Close to the left boundary forward interpolation must be used for cubic interpolation, resulting in

$$\tilde{f}_{j+1,1} = \frac{5f_{j,0} + 15f_{j,1} - 5f_{j,2} + f_{j,3}}{16} \tag{1.17}$$

and on the right boundary a backward stencil must be used as follows

$$\tilde{f}_{j+1,2^{j+1}-1} = \frac{f_{j,2^j-3} - 5f_{j,2^j-2} + 15f_{j,2^j-1} + 5f_{j,2^j}}{16} \tag{1.18}$$

With the use of cubic interpolation, i.e., four point interpolation stencil ($st = 4$) it is not possible to go lower than $j = 3$ during direct and inverse wavelet transform. The pseudo-code for direct wavelet transform with linear interpolation of a given function $f(x_i)$ over $i = 0, \ldots, 2^J$ points, excluding boundary points is given in the following:



(a) In physical space (b) Modulus of its Fourier transform

Figure 1.3: The scaling function $\Phi_{1,i}$ of wavelet transform corresponding to cubic interpolation.

1. DO for $j = J$ to $j = 1$, $step = -1$
2. $Pointer := 2^{J-j}$
3. DO for $i = Pointer$ to $i = 2^J - Pointer$, $step = 2 \times P$
4. $Interpol := 0.5f(i - Pointer) + 0.5f(i + Pointer)$
5. $d(i) := f(i) - Interpol$
6. END DO
7. END DO

We can note that no intermediate memory is needed in this transformation and the order of operations is $O(cN)$, where N is the number of points and the constant c depends on the order of interpolation which is usually less than 10. The pseudo-code for inverse transformation is given in the following:

```
1. DO  for  j = 1  to  j = J,  step = 1
2.     Pointer := 2^{J-j}
3.     DO  for  i = Pointer  to  i = 2^J - Pointer,  step = 2 × P
4.     Interpol := 0.5 f(i - Pointer) + 0.5 f(i + Pointer)
5.     f(i) := d(i) + Interpol
6.     END DO
7. END DO
```

1.2.2 Filtering in wavelet space

Given a threshold parameter for the finest ϵ_J or the coarsest ϵ_1 level, data compression will be obtained by thresholding of the detail coefficients, also called nonlinear filtering, applied to wavelet coefficients in wavelet space. After performing the direct transform, wavelet coefficients smaller than a threshold are set to zero and the corresponding point can be eliminated from the set of the points, in other words we can find the value of that point by interpolation and the error remains bounded by the threshold value.

$$d_{j,i} = \begin{cases} 0 & \text{if } |d_{j,i}| \leq \epsilon_j, \\ d_{j,i} & \text{else} \end{cases} \tag{1.19}$$

where

$$\epsilon_j = \epsilon_J \; 2^{D(j-J)} = \epsilon_0 \; 2^{D(j)} \tag{1.20}$$

in which $D = 1, 2, 3$ is the dimension of the problem, and J denotes the maximum level. A linear filtering also can be used, i.e., all wavelet coefficients above a given scale are set to zero. In the present investigation nonlinear filtering is used. After nonlinear filtering in wavelet space the given function $f(x)$, can be reconstructed $\overline{f}(x)$, just with the significant wavelet coefficients corresponding to the important points of the function. Those points must be kept to guaranty the boundedness of the error introduced due to filtering and eliminating non necessary points. Following Donoho [17], it can be shown that for a sufficiently smooth function $f(x)$

$$|f(x) - \overline{f}(x)| \leq c_1 \epsilon_0 \tag{1.21}$$

This implies that the number of significant wavelet coefficients N_s is bounded by ϵ_0 as

$$N_s \leq c_2 \epsilon_0^{-D/P_{WT}} \tag{1.22}$$

where P_{WT} is the order of the wavelet, i.e, the number of neighboring points, or the stencil size, used for wavelet construction [22] and the coefficients c_i depend on $f(x)$

(but are of order unity). Note that P_{WT} controls the number of zero moments of the interpolating scaling function. Combining Eq. (1.21) and Eq. (1.22) we have the following bounded error in terms of N_s

$$|f(x) - \overline{f}(x)| \leq c_3 N_s^{-P_{WT}/D} \tag{1.23}$$

The estimated error is consistent with numerical experiments for both one-dimensional and two-dimensional cases [37]. The accuracy of differentiation with wavelet transform applied to a function over a uniform and thresholded grid was examined by Vasilyev and Bowman [27] for the one dimensional case and by Vasilyev [34] in multiple dimensions. It was shown that the error bound on the derivative is given by

$$|D_x^{Uni} f(x) - D_x^{MR}\, \overline{f}(x)| \leq c_4 N_s^{-(P_{WT}-1)/D} \approx c_4 \epsilon_0^{(P_{WT}-1)/P_{WT}} \tag{1.24}$$

where D_x stands for the derivative operator in the x direction. This relation was verified numerically for both one and two dimensions in [27] and [34]. Note that the error bound (1.24) is also correct for the second-order derivative if a symmetric stencil is used [37].
The discrete L_p-norm for error estimation is defined by

$$L_p(f, \overline{f}) = \Big[\frac{1}{N}\sum_{i=1}^{N} |f_i - \overline{f}_i|^p\Big]^{\frac{1}{p}} \tag{1.25}$$

and L_∞-norm is defined as

$$L_\infty(f, \overline{f}) = \max |f_i - \overline{f}_i|,\ i = 1, \ldots, N. \tag{1.26}$$

The compression rate can be introduced as

$$\%\,\text{Compression} = \Big(\frac{\mathrm{N} - \mathrm{N_s}}{\mathrm{N}}\Big) \times 100 \tag{1.27}$$

where N is the number of total points and N_s is the number of significant points after thresholding in wavelet space. The idea of filtering and compression with error analysis will be shown for two examples.

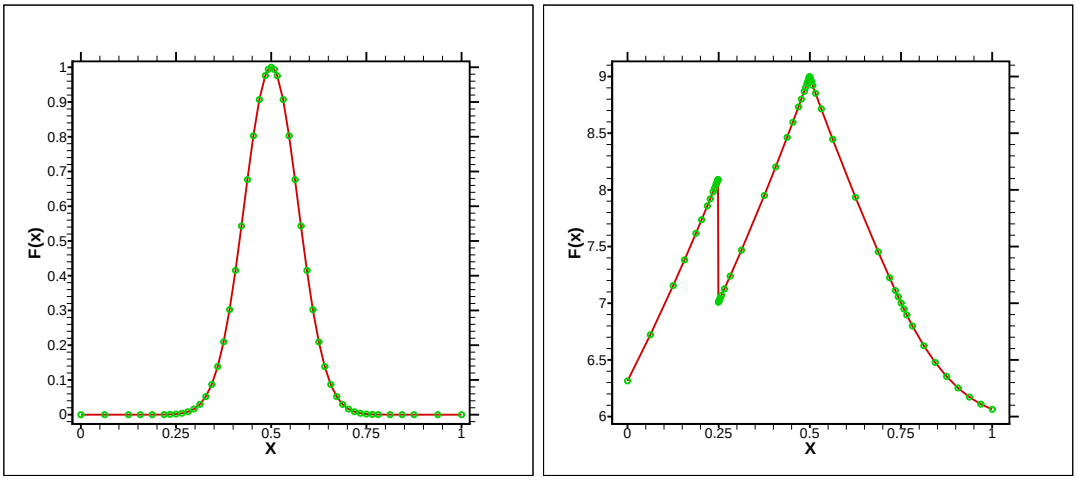
Example: Consider a non-periodic one-dimensional function $f(x)$ over $[0, 1]$

$$f(x) = \begin{cases} 8.1e^{1/4}e^{-|x-1/2|} & 0.0 \leq x < 0.25 \\ 9e^{-|x-1/2|} & 0.25 \leq x < 0.75 \\ e^{-|x-1/2|}(16x^2 - 24x + 18) & 0.75 \leq x \leq 1.0 \end{cases} \tag{1.28}$$

with a discontinuity at $x = 0.25$, a discontinuity in the first derivative at $x = 0.5$ and a discontinuity in second derivative at $x = 0.75$. As a second example we consider a Gaussian function

$$f(x) = \exp\Big(\frac{x - 0.5}{\delta}\Big)^2 \quad \text{for} \quad x \in [0, 1]$$

their sparse point representations, with the use of cubic ($P_{WT} = 4$) interpolating wavelet transform, for $J = 10$, filtered with threshold $\epsilon = 1 \times 10^{-3}$ are illustrated in Fig. 1.4 (a) and (b). Results of the error analysis and compression rates with the linear and cubic interpolating wavelet transforms, by applying different thresholding parameters are reported in Tables 1.1 and 1.2.



(a) Gaussian function, $\delta = 0.1$, Compression = 95%, L_∞-Error $\leq 1 \times 10^{-4}$

(b) Function (1.28), Compression = 94%, L_∞-Error $\leq 5 \times 10^{-5}$

Figure 1.4: Sparse point representation of 1D functions, obtained by WT with cubic interpolation ($J = 10$), filtered with threshold $\epsilon = 1 \times 10^{-3}$. The green dots (marked ●) show the retained grid points.

Table 1.1: Results of the error analysis and compression rates for the function (1.28) with the linear and cubic interpolating wavelet transforms, $J = 10$.

WT	ϵ	% Compression	L_∞-Error	L_1-Error	L_2-Error
Linear	1×10^{-1}	96.2	2×10^{-2}	1×10^{-3}	5×10^{-5}
Linear	1×10^{-2}	93.0	9×10^{-3}	2×10^{-4}	1×10^{-5}
Linear	1×10^{-3}	86.8	2×10^{-4}	6×10^{-5}	2×10^{-6}
Linear	1×10^{-4}	72.7	5×10^{-5}	1×10^{-5}	5×10^{-7}
Linear	1×10^{-5}	42.1	1×10^{-5}	2×10^{-6}	9×10^{-8}
Linear	1×10^{-6}	0.0	9×10^{-16}	3×10^{-17}	5×10^{-18}
Cubic	1×10^{-1}	96.3	7×10^{-2}	6×10^{-2}	1×10^{-4}
Cubic	1×10^{-2}	95.3	9×10^{-3}	5×10^{-5}	1×10^{-5}
Cubic	1×10^{-3}	94.0	5×10^{-5}	1×10^{-6}	1×10^{-7}
Cubic	1×10^{-5}	92.4	8×10^{-7}	9×10^{-8}	4×10^{-9}
Cubic	1×10^{-7}	88.7	2×10^{-8}	5×10^{-9}	2×10^{-10}
Cubic	1×10^{-9}	67.0	6×10^{-11}	2×10^{-11}	9×10^{-13}
Cubic	1×10^{-11}	36.7	3×10^{-12}	9×10^{-13}	4×10^{-14}
Cubic	1×10^{-13}	0.0	5×10^{-15}	9×10^{-16}	4×10^{-17}

Table 1.2: Results of the error analysis and compression rates for the Gaussian function, $\delta^2 = 0.1$, with the linear and cubic interpolating wavelet transforms, $J = 10$.

WT	ϵ	% Compression	L_∞-Error	L_1-Error	L_2-Error
Linear	1×10^{-1}	97.2	3×10^{-3}	9×10^{-4}	4×10^{-5}
Linear	1×10^{-2}	94.4	6×10^{-4}	2×10^{-4}	8×10^{-6}
Linear	1×10^{-3}	88.4	2×10^{-4}	5×10^{-5}	2×10^{-6}
Linear	1×10^{-5}	52.7	1×10^{-5}	2×10^{-6}	1×10^{-7}
Linear	1×10^{-6}	4.1	9×10^{-7}	2×10^{-8}	4×10^{-9}
Linear	1×10^{-7}	0.2	1×10^{-9}	3×10^{-12}	2×10^{-12}
Linear	1×10^{-8}	0.2	1×10^{-9}	3×10^{-12}	2×10^{-12}
Linear	1×10^{-10}	0.0	1×10^{-16}	6×10^{-18}	7×10^{-19}
Cubic	1×10^{-1}	96.3	7×10^{-2}	6×10^{-2}	1×10^{-4}
Cubic	1×10^{-2}	94.4	6×10^{-4}	2×10^{-4}	8×10^{-6}
Cubic	1×10^{-3}	96.8	3×10^{-5}	7×10^{-6}	3×10^{-7}
Cubic	1×10^{-4}	94.2	6×10^{-6}	8×10^{-7}	5×10^{-8}
Cubic	1×10^{-6}	87.6	1×10^{-8}	3×10^{-9}	1×10^{-10}
Cubic	1×10^{-8}	71.6	5×10^{-9}	1×10^{-9}	5×10^{-11}
Cubic	1×10^{-10}	12.1	1×10^{-10}	7×10^{-12}	7×10^{-13}
Cubic	1×10^{-12}	0.0	7×10^{-16}	9×10^{-17}	5×10^{-18}

1.2.3 Two-dimensional transform

The previously presented one-dimensional wavelet transform can be extended to higher dimensions. For the moment just the two-dimensional transform will be considered, since the three-dimensional transform can be constructed analogously. There are two ways to accomplish this [41]:

1. Rectangular wavelet transform
2. Tensor product wavelet transform

Rectangular wavelet transform : Like the Fourier transform for higher dimensions this transform is simply equivalent to applying the one dimensional wavelet transform to the rows and the columns of a matrix or a two-dimensional function.

$$f(x,y) = \sum_{j_x,k_x} \sum_{j_y,k_y} d_{j_x,k_x,j_y,k_y} \Psi_{j_x,k_x,j_y,k_y}(x,y) \tag{1.29}$$

where

$$\Psi_{j_x,k_x,j_y,k_y}(x,y) = \Psi_{j_x,k_x}(x)\Psi_{j_y,k_y}(y) \tag{1.30}$$

and

$$d_{j_x,k_x,j_y,k_y} = \langle f, \Psi_{j_x,k_x,j_y,k_y} \rangle \tag{1.31}$$

For some applications, this method is more advantageous. Often the notion of a scale has a certain meaning, so one would like to have a unique scale assigned to

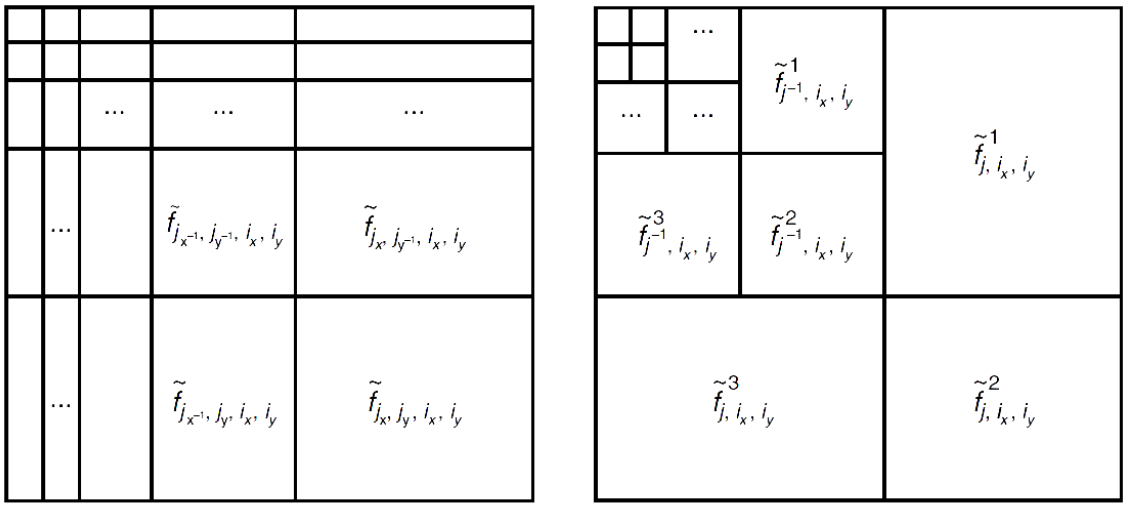


Figure 1.5: Schematic representation of two-dimensional wavelet transforms : rectangular WT (left), tensor product WT (right), picture taken from [41].

each basis function. For CFD applications the concept of multiresolution analysis (MRA) is much more interesting so the two-dimensional wavelet transform based on tensor product will be introduced. A schematic representation of the mentioned two-dimensional wavelet transforms is shown in Fig. 1.5.

Tensor product wavelet transform : This transform can be obtained through the tensor product of two one-dimensional multiresolution analysis of $L^2(R)$, so three different wavelets will be generated

$$f(x,y) = \sum_j \sum_{i_x,i_y} \sum_{\mu=1}^{3} d^{\mu}_{j,i_x,i_y} \Psi^{\mu}_{j,i_x,i_y}(x,y) \tag{1.32}$$

where

$$\Psi^{\mu}_{j,i_x,i_y}(x,y) = \begin{cases} \Psi_{j,i_x}(x)\Phi_{j,i_y}(y), & \mu = 1 \\ \Phi_{j,i_x}(x)\Psi_{j,i_y}(y), & \mu = 2 \\ \Psi_{j,i_x}(x)\Psi_{j,i_y}(y), & \mu = 3 \end{cases} \tag{1.33}$$

and

$$d^{\mu}_{j,i_x,i_y} = \langle f, \Psi^{\mu}_{j,i_x,i_y} \rangle = f_{j/j+1}(i_x,i_y) - \tilde{f}^{\mu}_{j/j+1}(i_x,i_y) \tag{1.34}$$

The scale parameter j simultaneously controls the dilatation in x and y directions, and in D dimensions this construction yields to $2^D - 1$ types of wavelets [41].

Application to interpolating wavelet transform : For a two-dimensional function $f(x,y)$ with the use of the notion of tensor product, the interpolating wavelet transform will take three different forms for locations with following (i_x, i_y) index pair, where $i_x = 1, \ldots, 2^{J+1}$ and $i_y = 1, \ldots, 2^{J+1}$ are the indexes in x and y directions, respectively [21].

(odd,odd): Will not change (they belong to the lower levels) see Fig. 1.6.

(even,odd): Interpolation along x direction, $\mu = 1$

$$\tilde{f}^1_{j+1}(2i_x, 2i_y + 1) = \sum_{n=1}^{st/2} \beta_n \Big(f_j(i_x + n - 1, i_y) + f_j(i_x - n, i_y) \Big) \tag{1.35}$$

(odd,even): Interpolation along y direction, $\mu = 2$

$$\tilde{f}^2_{j+1}(2i_x + 1, 2i_y) = \sum_{m=1}^{st/2} \beta_m \Big(f_j(i_x, i_y + m - 1) + f_j(i_x, i_y - m) \Big) \tag{1.36}$$

(even,even): 2D interpolation at the same level (diagonal direction), $\mu = 3$

$$\tilde{f}^3_j(2i_x, 2i_y) = \sum_{n=1}^{st/2} \beta_n \sum_{m=1}^{st/2} \beta_m \Big(f_j(i_x + n - 1, i_y + m - 1) + f_j(i_x - n, i_y + m - 1) + f_j(i_x + n - 1, i_y - m) + f_j(i_x - n, i_y - m) \Big)$$

In the above equations the coefficients β_i, $(i = 1, 2, \ldots, st/2)$ come from standard interpolation methods, where $(st - 1)$ is the order of the interpolation, (st being the stencil size). For a few common values of st we have

$st = 2$ (linear interpolation) with $\beta_1 = 0.5$

$st = 4$ (cubic interpolation) with $\beta_1 = \frac{9}{16}, \beta_2 = -\frac{1}{16}$

$st = 6$ (fifth-order interpolation) with $\beta_1 = \frac{150}{256}, \beta_2 = -\frac{25}{256}, \beta_3 = \frac{3}{256}$

The pseudo-code for the direct wavelet transformation, with linear interpolation, for a matrix or a two-dimensional function $f(x_{i_x}, y_{i_y})$, $i_x, i_y = 1, \ldots, N$, with $N = 2^J + 1$ points in each direction including boundary points, is given in the following:

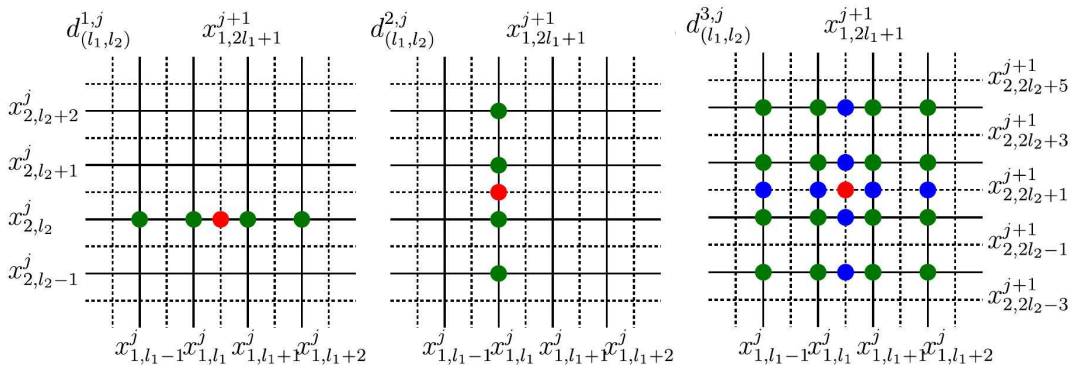
```
1. DO   for   j = J   to   j = 0,   step = −1
2.     P := 2^{J−j}
3.     DO   for   i_y = 1 + P   to   i_y = N − P,   step = 2 × P
4.     DO   for   i_x = 1 + P   to   i_x = N − P,   step = 2 × P
5.     Interpol := 0.25 × (f(i_x − P, i_y) + f(i_x + P, i_y) + f(i_x, i_y − P) + f(i_x, i_y + P))
6.     d(i_x, i_y) := f(i_x, i_y) − Interpol    END DO   END DO
7.     DO   for   i_y = 1   to   i_y = N,   step = 2 × P
8.     DO   for   i_x = 1 + P   to   i_x = N − P,   step = 2 × P
9.     Interpol := 0.5 × (f(i_x − P, i_y) + f(i_x + P, i_y))
10.    d(i_x, i_y) := f(i_x, i_y) − Interpol    END DO   END DO
11.    DO   for   i_y = 1 + P   to   i_y = N − P,   step = 2 × P
12.    DO   for   i_x = 1   to   i_x = N,   step = 2 × P
13.    Interpol := 0.5 × (f(i_x, i_y − P) + f(i_x, i_y + P))
14.    d(i_x, i_y) := f(i_x, i_y) − Interpol    END DO   END DO
15. END DO
```

The inverse transform will be similar by starting from the lowest level $j = 0$ up to $j = J$ and performing the two-dimensional interpolations after interpolations in x and y directions because in each level two-dimensional interpolations are using values from their own level. In Fig. 1.6 a schematic representation of two-dimensional wavelet transform based on cubic interpolation and tensor product is shown.

Example: Consider a two-dimensional function with a discontinuity

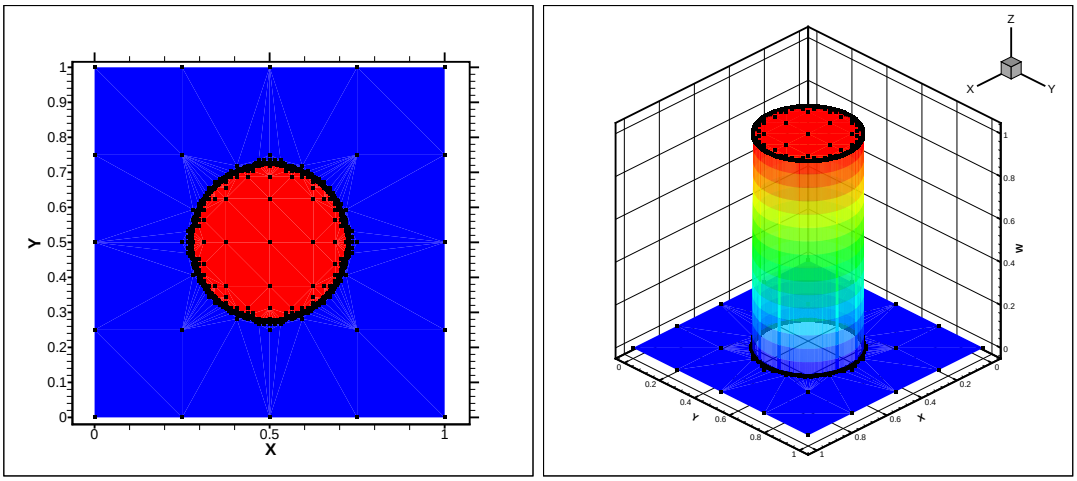
$$f(x,y) = \begin{cases} 1 & \text{if } (x-1/2)^2 + (y-1/2)^2 \leq 1 \\ 0 & \text{else} \end{cases} \tag{1.37}$$

illustrated in Fig. 1.7 (b) and a two-dimensional Gaussian illustrated in Fig. 1.8 (b). Their sparse point representation, with the cubic interpolating wavelet transform, for $J = 9$, filtered with $\epsilon = 1 \times 10^{-3}$ is also shown in Fig. 1.7 (a) and Fig. 1.8 (a). Results of error analysis and compression rates with the linear and cubic interpolating wavelet transforms, by applying different thresholding parameters are reported in Tables 1.3 and 1.4.



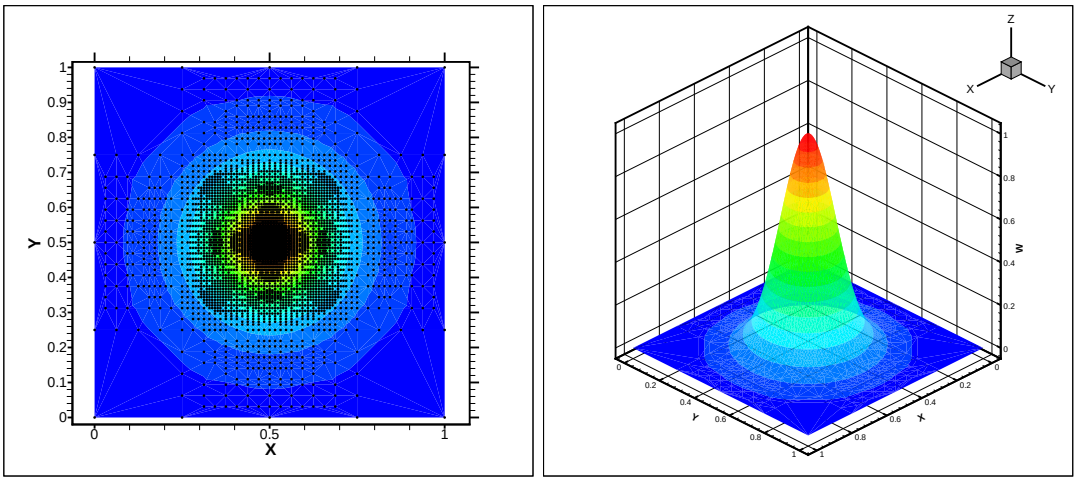
(a) Interpolation in x and y directions (b) 2D interpolation

Figure 1.6: Schematic representation of the two-dimensional wavelet transform with $p = 4$, for calculation of the wavelet coefficients $d^{\mu}_{j}(i_x, i_y)$, $\mu = 1, 2, 3$. In the one-dimensional interpolations in x and y directions, ($\mu = 1, 2$) points at the finer level $j + 1$ (marked •) must interpolated from the points at the coarser level j (marked •), illustrated in (a). In the two-dimensional interpolation, ($\mu = 3$) point at the level $j + 1$ (marked •) must interpolated from the points at the same level $j + 1$ (marked •), illustrated in (b), picture taken from [37].



(a) 2D view (b) 3D view

Figure 1.7: Function (1.37) and its sparse point representation (marked •) with the linear interpolating wavelet transform, $J = 9$, $\epsilon = 10^{-3}$, Compression = 99.4%, L_1-Error= 10^{-15}.



(a) 2D view (b) 3D view

Figure 1.8: Gaussian function and its sparse point representation (marked •) with the cubic interpolating wavelet transform, $J = 9$, $\epsilon = 10^{-3}$, Compression= 98%, L_1-Error= 1.14×10^{-4}.

Table 1.3: Results of the error analysis and compression rates for the (discontinues) function (1.37) with the linear and cubic interpolating wavelet transforms, $J = 9$.

WT	ϵ	% Compression	L_∞-Error	L_1-Error	L_2-Error
Linear	1×10^{-1}	99.4	10^{-15}	10^{-15}	10^{-15}
Linear	1×10^{-7}	99.4	10^{-15}	10^{-15}	10^{-15}
Linear	1×10^{-13}	99.4	10^{-15}	10^{-15}	10^{-15}
Cubic	1×10^{-1}	98.2	10^{-15}	10^{-15}	10^{-15}
Cubic	1×10^{-7}	98.2	10^{-15}	10^{-15}	10^{-15}
Cubic	1×10^{-13}	98.2	10^{-15}	10^{-15}	10^{-15}

Table 1.4: Results of the error analysis and compression rates for the two-dimensional Gaussian function with the linear and cubic interpolating wavelet transforms, $J = 9$.

WT	ϵ	% Compression	L_∞-Error	L_1-Error	L_2-Error
Linear	1×10^{-1}	87.6	4×10^{-2}	9×10^{-3}	3×10^{-5}
Linear	1×10^{-2}	58.8	6×10^{-3}	2×10^{-3}	4×10^{-6}
Linear	1×10^{-3}	37.2	5×10^{-4}	1×10^{-4}	3×10^{-7}
Linear	1×10^{-5}	7.7	2×10^{-6}	2×10^{-7}	1×10^{-9}
Linear	1×10^{-8}	0.0	3×10^{-16}	4×10^{-18}	3×10^{-20}
Cubic	1×10^{-2}	99.8	6×10^{-3}	1×10^{-3}	4×10^{-6}
Cubic	1×10^{-3}	98.4	9×10^{-4}	1×10^{-4}	4×10^{-7}
Cubic	1×10^{-5}	58.1	4×10^{-6}	3×10^{-7}	1×10^{-9}
Cubic	1×10^{-8}	6.1	6×10^{-11}	4×10^{-13}	5×10^{-15}
Cubic	1×10^{-13}	0.0	6×10^{-16}	2×10^{-17}	9×10^{-20}

Chapter 2

An adaptive multiresolution method for evolutionary PDEs

2.1 Introduction

In this chapter first the theoretical basis of an adaptive multiresolution method for evolutionary PDEs will be explained in some detail and two time integration methods from the Runge-Kutta family will be presented. Then the space discretization based on finite differences on Cartesian grids coupled with wavelet optimized grid (WOG) adaptation will be described. Finally, the method will be extended to the two-dimensional Navier-Stokes equations in vorticity and stream-function formulation including the penalization term which imposes no-slip and no-penetration boundary conditions, required in the case of complex geometries, e.g., curved walls. Penalization methods allow to impose boundary conditions in the case of complex geometries (even varying in time) without modifying the computational grid.

2.2 Evolutionary PDEs

In fluid mechanics we encounter different types of partial differential equations. They come from conservation laws and each of them describes the evolution of a different quantity, i.e., momentum, energy, vorticity and etc. From mathematical point of view these equations can be divided into three main groups, i.e., hyperbolic, parabolic and elliptic. The Euler equations are an example of a hyperbolic system of equations. Two examples for parabolic equations are :
the one-dimensional viscous Burgers equation

$$\partial_t u + u u_x = \nu u_{xx} \tag{2.1}$$

and the two-dimensional vorticity ($\omega = \nabla \times \mathbf{u}$) transport equation

$$\partial_t \omega + (\mathbf{u} \cdot \nabla)\omega = \nu \nabla^2 \omega. \tag{2.2}$$

As an example for elliptic PDEs, Poisson equation

$$-\nabla^2 \psi = \omega \tag{2.3}$$

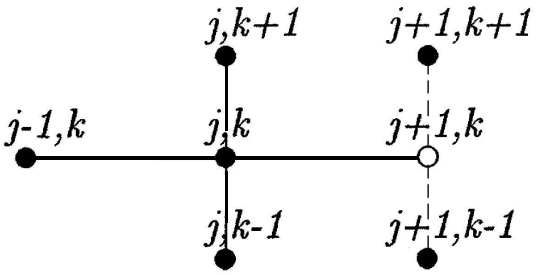


Figure 2.1: Hung point (marked ∘) and active points (marked •), picture taken from [32].

for stream function ψ can be cited. These equations must completed with suitable initial and boundary conditions. In unsteady incompressible flows the governing system of equations, i.e., Navier-Stokes equations, have parabolic character in time (for steady problems they have elliptic character in space). The continuity equation, must be satisfied in all time steps so the evolutionary part of the problem will be the momentum (or vorticity) equations. Hence for instance, we will focus on the time dependent equations in the form of :

$$\frac{\partial u}{\partial t} = RHS \tag{2.4}$$

where RHS contains all spatial terms either linear or nonlinear. It is also possible to write the equations in conservative form

$$\frac{\partial u}{\partial t} = \nabla \cdot F(u) \quad \text{on } \Omega \tag{2.5}$$

where F contains all viscous and inviscid fluxes.

2.3 Spatial discretization

For discretization of spatial derivatives a second order central finite difference method over Cartesian grids will be used as follows:

First derivative:

$$\frac{\partial u}{\partial x}\Big|_i = \frac{u_{i+1} - u_{i-1}}{2h} + O(h)^2 \tag{2.6}$$

Second derivative:

$$\frac{\partial^2 u}{\partial x^2}\Big|_i = \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} + O(h)^2 \tag{2.7}$$

For diffusion terms with a variable coefficient λ we have:

$$\frac{\partial}{\partial x}\Big(\lambda \frac{\partial u}{\partial x}\Big)\Big|_i = \frac{(\lambda_{i+1} + \lambda_i)(u_{i+1} - u_i) - (\lambda_{i-1} + \lambda_i)(u_i - u_{i-1})}{2h^2} + O(h)^2 \tag{2.8}$$

For maintaining the accuracy in the multiresolution analysis by using symmetric

stencils the indices (± 1) are replaced by ($\pm d$), where d is an integer number, multiplying by h, representing the distance (dh) between the point in i and the nearest active point. For example Eq. (2.7) will be replaced by

$$\frac{\partial^2 u}{\partial x^2}\Big|_i = \frac{u_{i+d} - 2u_i + u_{i-d}}{(dh)^2} + O(h)^2 \tag{2.9}$$

For a given grid point after finding the nearest active point, all neighbors will be checked with the same distance to verify if they are in the list of active points, if not, we will call them hung points and their values must be interpolated from active points in lower levels, see Fig. 2.1. For two-dimensional problems the same relations will be applied to x and y directions.
Another alternative to the idea of the hung point interpolations is the use of non-symmetric stencils. In this approach evaluation of the first derivative via a non-symmetric stencil reads:

$$\frac{\partial u}{\partial x}\Big|_i = \frac{a^2 u_{i+b} + (b^2 - a^2)u_i - b^2 u_{i-a}}{ab(a+b)h} + O(h) \tag{2.10}$$

Second derivative can evaluated by:

$$\frac{\partial^2 u}{\partial x^2}\Big|_i = \frac{2au_{i+b} - 2(a+b)u_i + 2bu_{i-a}}{ab(a+b)h^2} + O(h) \tag{2.11}$$

where a and b are integer numbers, multiplied by h, they represent the distances from left and right neighbors respectively [26].

2.4 Time integration

In this section the time integration for a general evolutionary PDE is described and the results will be assessed in the next chapter. For the moment just the explicit Rung-Kutta family is considered. A classical fourth order Runge-Kutta method with fixed time step and an adaptive time stepping with embedded Runge-Kutta methods will be presented.

2.4.1 Explicit fourth order Runge-Kutta method

Because of high accuracy and straightforward parallelization, the fourth order Runge-Kutta method is one of the best and mostly used methods for integration of ordinary differential equations [18]. By putting all discretized spatial derivatives in the RHS operator one can solve the considered partial differential equation as a system of ordinary differential equations in time. The RHS can be interpreted as the slope of the value of the function in time. This method consists of four stages, including two predictor and two corrector steps, see Fig. 2.2.

First step:

$$k_1 = RHS(u)$$

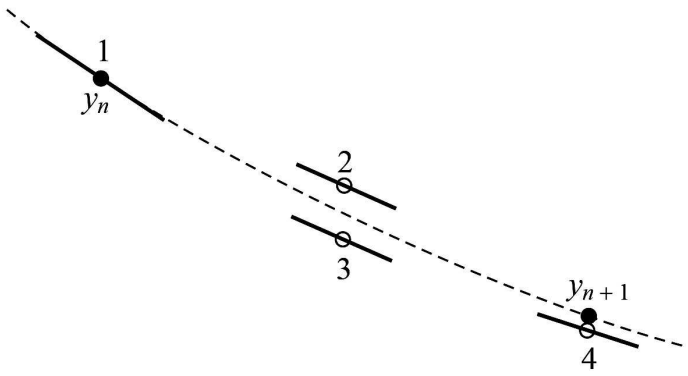


Figure 2.2: Schematic representation of the fourth-order Runge-Kutta method. In each time step the RHS operator must be evaluated four times: once at the initial point (marked $\bullet$), twice at trial midpoints (marked $\circ$) and once at a trial endpoint (marked $\circ$). From these derivatives the value of the function in the next time step (marked $\bullet$) can be calculated, picture taken from [18].

$$u^* = u^n + \frac{\Delta t}{2} k_1 \tag{2.12}$$

Second step:

$$k_2 = RHS(u^*)$$
$$u^* = u^n + \frac{\Delta t}{2} k_2 \tag{2.13}$$

Third step:

$$k_3 = RHS(u^*)$$
$$u^* = u^n + \Delta t \; k_3 \tag{2.14}$$

Fourth step:

$$k_4 = RHS(u^*)$$
$$u^{n+1} = u^n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4) \tag{2.15}$$

Usually in addition to one memory location for u, five memory locations are necessary at each grid point for the evaluation of k_1, k_2, k_3, k_4 and u^*. However it is possible to release k_2 after the evaluation of u^* in the third step, by simply adding the value of k_2 to k_3, say

$$\tilde{k_3} \leftarrow k_3 + k_2$$

So it is possible to use again the allocated memory k_2 instead of k_4 during the computation. Hence the fourth step will change into

$$\tilde{k_2} = RHS(u^*)$$

$$u^{n+1} = u^n + \frac{\Delta t}{6}(k_1 + 2\tilde{k_3} + \tilde{k_2}).$$

2.4.2 Adaptive time stepping with embedded Runge-Kutta methods

For time integration of the considered ODE, instead of using a fixed time step Δt chosen a priori, it can be advantageous to have a technique that automatically adjusts its size dynamically to advance faster in time when possible and to guarantee the error in solution is bounded by a given value [50]. The time step Δt must be chosen sufficiently small to satisfy a required precision of the computed results, denoted by δ_{desired}, but it must be sufficiently large to avoid unnecessary computational work. Typically, if u^1 is the approximation of $u^{\text{exact}}(t+\Delta t)$ estimated by a method of order p, asymptotic developments of the local truncation errors of the form

$$u^{\text{exact}}(t+\Delta t) - u^1 = C_1 \Delta t^p + O(\Delta t^{p+1}) \tag{2.16}$$

can be used to find the step size required to attain a specified accuracy. However, since the leading constant C is not known a priori, practical error estimates are necessary. To estimate the local truncation error, the idea is to apply two embedded ODE solvers, one with order p and the other with order $p+1$. If u^2 is the approximation of $u(t+\Delta t)$ estimated by a method of order $p+1$, then, for sufficiently small Δt we have

$$u^1 - u^2 \approx -C_1 \Delta t^p + C_2 \Delta t^{p+1} \approx -C_1 \Delta t^p \tag{2.17}$$

this yields to estimate

$$C_1 \approx \frac{u^2 - u^1}{\Delta t^p} = \frac{\delta}{\Delta t^p} \tag{2.18}$$

By choosing a RK2 and RK3 method for approximations one gets

$$u^* = u^n + \Delta t \ RHS(u^n)$$

$$u^{RK2} = \frac{1}{2}(u^n + u^* + \Delta t \ RHS(u^*)) \tag{2.19}$$

and

$$u^{**} = \frac{1}{4}(3u^n + u^* + \Delta t \ RHS(u^*))$$

$$u^{n+1} = u^{RK3} = \frac{1}{3}(u^n + u^* + 2\Delta t \ RHS(u^{**})) \tag{2.20}$$

where

$$\delta = ||u^{p+1} - u^p||_\infty = ||u^{RK3} - u^{RK2}||_\infty$$

Hence the step size required to maintain the local truncation error of the first scheme below δ_{desired} is given by

$$\Delta t_{new} = \xi \Delta t_{old}$$

where

$$\xi = \left[\frac{\delta_{\text{desired}}}{\delta}\right]^{\frac{1}{p}}$$

For RK2/3 one chooses $p = 2$. To prevent the time step from varying too abruptly or to be sure that Δt_{new} in fact will produce an error less than δ_{desired}, some care is needed. In this method, we cannot go back to the previous time step once the solution at the new time step is computed due to the low storage memory model we are using. Hence it is better to limit the increase of the time step by introducing a so-called safety factor S. The new time step Δt_{new} is chosen such that

$$\xi = \begin{cases} 1+S & \xi > 1+S \\ \xi & 1-S > \xi > 1+S \\ 1-S & \xi < 1-S \end{cases} \tag{2.21}$$

where S depends to the problem and must be chosen in a way to not destabilize the time integration, e.g., $S = 0.001$. Moreover one memory location for u, four memory locations are needed in each grid point [54].

2.5 Multiresolution analysis

In the following, the multiresolution algorithm for a typical evolutionary PDE is briefly summarized. For more details on the general adaptive MR algorithm see [54]. Our multiresolution analysis is based on the cubic interpolating wavelet transform, described in some details in chapter 1. Because of the change in the grid points during each time step, the MR method is not applicable for backward multi step methods in time like Adams-Bashforth, instead, one must choose forward multi step methods like Runge-Kutta family for the time integration of the given PDE. First, depending on the regularity of the solution from previous time step (or the initial condition), we have the values (or the initial values) of an arbitrary dependent variable over a dyadic grid. According to the regularity of the solution, some points have an updated value, we call them *active points*. To perform a wavelet transform over the active points, we suppose the arrangement of these points over the dyadic grid is in such a manner that by starting from the finest level, down to the coarsest level, all necessary points are in the list of the active points. The wavelet transform is performed to compute the wavelet coefficients of these active points, which are also called details. Afterwards thresholding (nonlinear filtering) of the details (wavelet coefficients) is performed to determine the new active points (i.e., retained grid points are in a one-to-one correspondence with significant wavelet coefficients). Next a so-called safety zone, which will be described in more details further down, must be added to the new active points. As regards the values of some newly added points are unknown, an inverse wavelet transform will be performed just for the points that their values are unknown, that is to say, their values will be interpolated from known points in lower levels by performing an inverse wavelet transform with $d_{j,i} = 0$. After that spatial derivatives in RHS must be evaluated with the use of the equations described in Section 2.3. Finally, a time evolution is made just for the active points.
The threshold operator (nonlinear filtering) depending on the choice of the threshold ϵ_0, will select the important points for obtaining compression, while keeping the error introduced by filtering of wavelet coefficients (grid adaptation) below the truncation

error of the spatial discretization scheme.

Safety zone

The presence of nonlinear terms in the governing equations of fluid mechanics will imply the creation of smaller scales in the flow, which may lead to energy transfer into the finer scales. Therefore when solving the governing evolutionary PDEs with MR technique, one needs to apply an additional criterion to ensure that the wavelet basis corresponding to a computational mesh is sufficient to approximate the solution throughout the time-integration stage for an evolution problem. To ensure the adequate approximation of the solution during time integration, Liandrat & Tchamitchian (1990) introduced the concept of a safety or an adjacent zone, which adds the points whose wavelet coefficients can possibly become significant during the period of time integration to the current significant points, when the grid remains unchanged. Most current wavelet-adaptation techniques are based on the concept of *safety zone*. In actual implementations, the safety zone (neighboring wavelets at the same and one above levels of resolution) is added for each significant coefficient. The sufficiently small time step usually chosen by explicit time integration methods, e.g., Runge-Kutta methods, ensures that the regularity of the solution does not propagate outside the safety zone. This typically results in a CFL-like (cell-Reynolds) constraint (Courant et al. 1928), which ensures that no energy at a given resolution scale propagates outside the safety region. The thickness of the safety zone determines the time interval during which the calculations can be carried out without modifying the computational grid. However, for computational efficiency, it was found (e.g., Schneider et al. 2006, Vasilyev 2003) that the safety zone, which includes the immediate neighboring wavelets, is the optimal [52]. The adaptation strategy for an evolution problem is illustrated in Fig. 2.3. In our simulations the number of points in the safety zone is almost equal to the number of active points.

As explained to allow the basis (grid points) to change, we have to extend the sparse point representation after thresholding. First, in space, we add neighbor points corresponding to neighbor wavelets in the sparse point representation. The number of points to be added depends also on the PDEs wave propagation speed. Then the grid points are able to adjust when the solution is moving in space. Second, in scale, we add neighbor points corresponding to wavelets on the next finer scale in the sparse point representation. The number of points to be added in upper scales depends to order of nonlinear term in the governing PDE, e.g., second order in the case of Navier-Stokes equations and third order in the case of nonlinear Schrödinger equation (NLS, discovered in optics and water waves). This refinement allows the accurate development of the solution in time and space, i.e., four points in one dimension and 16 points in two dimensions must be added but most of them will be common [24].

Moreover to the points added in the above procedure another check must be performed to add all necessary points to the current list of the active points for having a consistent direct or inverse WT. This depends on the stencil of the WT, we have presented two methods of interpolation in previous chapter, i.e., linear and cubic interpolation. In one dimension two points for linear interpolation and four points for cubic interpolation are necessary. For two-dimensions depending on the index

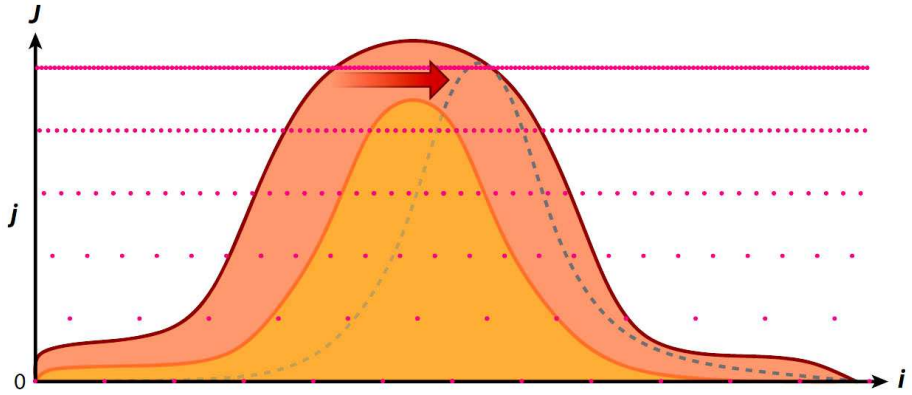


Figure 2.3: Illustration of the adjacent zone for adaptive wavelet methods in wavelet coefficient space with level (scale) index j and position index i. Solid circles (marked •) indicate the positions of wavelets. The locations of wavelets with significant coefficients kept after thresholding at time t are marked by the orange bell curve (——). The second bell-shaped region $(- - -)$ on the right of the original one marks the locations of the significant wavelet coefficients at the end of the time integration, i.e., at $t + \Delta t$. The safety zone is represented by red bell-shaped region, picture taken from [52].

composition (odd-even), different situations can occur, see Fig. 1.6. This step will determine all the points that will be used in the current time step to evaluate the solution.
Another point to be considered is the possibility to have, big jumps in the stencil of central finite differences in an adaptive method which may lead to a divergence of the value of the derivatives in some points. To prevent this problem, in each solution point the ratio of the distances of right and left neighbor points must not be more than two. In some literature this is called gradedness of the grid points and maybe it is equivalent to the check for wavelet interpolation stencil of WT based on linear interpolation. Of course the stencil of WT based on cubic interpolation is wider than linear interpolation and so the gradedness is guaranteed. This point will be optional if necessary points for interpolation via WT are already added to the active points.

Multiresolution analysis in summary
Denoting by $E(\Delta t)$ the discrete time evolution operator, the global algorithm can schematically be summarized by

$$u^{n+1} = E(\Delta t)\Big[M^{-1} \cdot S \cdot T(\epsilon) \cdot M\Big]u^n \tag{2.22}$$

where M and M^{-1} are the direct (WT) and inverse (IWT) wavelet transform operators. $T(\epsilon)$ is the thresholding operator and S represents the safety zone operator. For an Euler explicit time integration we have

$$E(\Delta t)u^n = u^n + \Delta t\, RHS(u^n).$$

where RHS operator contains all the terms of the considered evolutionary equation to be integrated except time derivative. In summary the algorithm is given in the following:

1. Start from an initial condition over a dyadic grid
2. Apply WT to the active points (from the finest level down to the coarsest level) to compute the wavelet coefficients of the independent variable
3. Perform thresholding $T(\epsilon)$ to remove all the points from the list of the active points which their wavelet coefficients are below the corresponding threshold ϵ_j
4. Add safety zone to the list of the new active points
   (a) Add neighbor points at the same and one above levels
   (b) Guarantee the gradedness of the new active points (optional)
   (c) Add necessary points to the current list of the active points, for having a consistent direct or inverse WT
5. Apply IWT to the new active points to compute the values of the independent variables (or interpolate the values of all newly added points via IWT with $d = 0$)
6. Perform the time evolution of the independent variable for all the active points
   (a) Search for the nearest active point to determine $dist$ for all active points
   (b) Check for the existence of all other neighbors of the active points with distance $dist$, mark all the missing points as the hung points
   (c) Interpolate the values of the hung points via IWT with $d = 0$
   (d) compute the spatial derivatives for the given PDE via FDM with symmetric stencils
7. Go to step 2, if $T < T_{end}$

N.B. 1 Before interpolation of the values of an independent variables via IWT (from the coarsest level up to the finest level) in some grid points (with wavelet coefficients equal to zero, $d = 0$), it is necessary to mark all the intermediate necessary points for having a consistent WT, (from the finest level down to the coarsest level) and adding them to the list of the points to be interpolated.

N.B. 2 For multi-step methods in time integration such as Runge-Kutta family before calculation of spatial derivatives at intermediate steps, the value of u^* for the hung points, must be interpolated again from the new values of active points. But (a) and (b) will be done once in each time step.

N.B. 3 In the case of the two-dimensional Navier-Stokes equations in velocity-vorticity formulation, before calculation of the spatial derivatives it is necessary to solve an elliptic equation, i.e., Eq. (2.27) for updating stream function.

2.6 Extension to two-dimensional incompressible Navier-Stokes equations

The governing equations of incompressible flows are the Navier-Stokes equations (2.23 - 2.24). In primitive variables momentum equations read

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\frac{1}{\rho_f}\nabla p + \nu \nabla^2 \mathbf{u} + \mathbf{F} \quad (x,t) \in \Omega \times [0,\infty) \tag{2.23}$$

and the continuity equation corresponds to

$$\nabla \cdot \mathbf{u} = 0 \quad , \quad \mathbf{x} \in \Omega \tag{2.24}$$

where Ω is the spatial domain of interest, given as an open subset of $\mathbb{R}^3$, which can be bounded or unbounded in general, $\mathbf{u}(\mathbf{x},t)$ is the velocity field which assumed to be at least in C^2 with respect to space and in C^1 with respect to time, $p(\mathbf{x},t)$ is the pressure field, $\nu = \mu/\rho_f$ is the kinematic viscosity, ρ_f is the fluid density and $\mathbf{F}(\mathbf{x},t)$ is a forcing term. Thus, for a complete description of a particular problem, the above equations need to be completed to describe an initial/boundary value problem (IBVP). Hence by specifying an initial condition $\mathbf{u}(\mathbf{x},0) = \mathbf{u}_0(\mathbf{x})$, which we assume to be in C^∞ and divergence free in all of Ω and by giving boundary conditions for velocity and pressure satisfying the global mass conservation constraint one will seek the solution during time evolution [38]. By choosing respectively U_∞ and L as reference velocity and length for a given problem the Navier-Stokes equations can be written in non-dimensional form in which $Re = U_\infty L/\nu$ is the Reynolds number and ρ is set to unity:

$$\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla)\mathbf{u} = -\nabla p + \frac{1}{Re}\nabla^2 \mathbf{u} + \mathbf{F} \tag{2.25}$$

However in two-dimensional problems as explained earlier the vorticity and stream-function formulation has the advantage that it not only eliminates the pressure variable entirely, but also ensures a divergence-free velocity field (continuity equation), if Eq. (2.27) properly satisfied. One will encounter with two scalar quantity, i.e., ψ and ω, instead of velocity vector and the pressure field, thus it makes the computations very efficient. We continue with this formulation, but the concepts are applicable also to primitive variable formulation. By taking the curl of Eq. (2.23) after elimination of terms due to two-dimensional assumption, incompressibility Eq. (2.24) and simplifications due to constant density, one obtains the vorticity transport equation for two-dimensional flows

$$\partial_t \omega + (\mathbf{u} \cdot \nabla)\omega = \nu \nabla^2 \omega + \nabla \times \mathbf{F} \tag{2.26}$$

where $\omega = \nabla \times \mathbf{u} = v_x - u_y$ denotes the vorticity. The equation is a parabolic equation in time and the velocity components are determined from $\mathbf{u} = (u,v) = (\partial_y \psi, -\partial_x \psi)$ with ψ being the stream function, satisfying

$$-\nabla^2 \psi = \omega \tag{2.27}$$

which is an elliptic[1] equation in space and will be solved numerically via iterative methods like successive over relaxation or multigrid methods. With the use of auxiliary relations for velocity components it is possible to eliminate velocity vector from Eq. (2.26) and obtain

$$\frac{\partial \omega}{\partial t} + \frac{\partial \psi}{\partial y}\frac{\partial \omega}{\partial x} - \frac{\partial \psi}{\partial x}\frac{\partial \omega}{\partial y} = \nu\Big(\frac{\partial^2 \omega}{\partial x^2} + \frac{\partial^2 \omega}{\partial y^2}\Big) + \Big(\frac{\partial F_y}{\partial x} - \frac{\partial F_x}{\partial y}\Big) \tag{2.28}$$

Boundary conditions

The boundary conditions for a curved boundary denoted by (s) moving tangent to its surface with a constant velocity U_{tan} can be written in terms of the stream function ψ, at each boundary section Γ_i. The no-penetration boundary condition is equivalent to

$$\frac{\partial \psi}{\partial \hat{\tau}}\Big|_{wall} = \hat{n} \cdot \mathbf{u}(s,t) = 0 \quad \text{(Neumann)} \quad \leftrightarrow \quad \psi|_{wall} = C_i \quad \text{(Dirichlet)} \tag{2.29}$$

and the no-slip boundary condition read

$$\frac{\partial \psi}{\partial \hat{n}}\Big|_{wall} = -\hat{\tau} \cdot \mathbf{u}(s,t) = \pm U_{tan} \quad \text{(Neumann)} \tag{2.30}$$

where C_i is a constant for each (Γ_i), $\hat{\tau}$ is the tangent to the wall direction and $\hat{n}$ is the direction normal to the wall. Free-slip boundary condition is evident and will achieved by $\omega = 0$. For a wall with zero tangential velocity we will have $U_{tan} = 0$. In a simply-connected domain, C_0 can be taken equal to zero ($C_0 = 0$). As a result for a fixed horizontal wall the no-penetration and no-slip boundary conditions are $(v = u = 0)$ or $(\psi_x = \psi_y = 0)$.

For simulation of two dimensional incompressible flows, beside the mentioned advantages of vorticity and stream-function formulation one can see some disadvantages. Yet, the main difficulties in the numerical implementation of this formulation come from the boundary conditions [31] the majors among them are as follows:

1. The implementation of the two cited boundary conditions for the stream function ψ simultaneously.

2. For updating the vorticity ω during time integration, there is no definite boundary condition for vorticity.

3. Determining the constants C_i at each boundary of "holes" Γ_i if the computational domain is multi-connected.

However, several methods have been proposed to update the vorticity boundary condition over a solid boundary moving tangent to its surface with a constant velocity U_{tan}. In the following we cite some of them, which will lead to second and fourth order accuracies :

[1] Perturbations will spread in all directions with the speed of sound which approaches to infinity in the incompressible limit.

Thom's formula for the vorticity at wall

$$\omega_{i,0} = \frac{\psi_{i,0} - \psi_{i,1} \pm hU_{tan}}{0.5h^2} + O(h^2) \tag{2.31}$$

Wilkes, Roach, Pearson and Jensen's formula for the vorticity at wall

$$\omega_{i,0} = \frac{7\psi_{i,0} - 8\psi_{i,1} + \psi_{i,2} \pm 6hU_{tan}}{2h^2} + O(h^2) \tag{2.32}$$

Briley's fourth order formula for the vorticity at wall

$$\omega_{i,0} = \frac{85\psi_{i,0} - 108\psi_{i,1} + 27\psi_{i,2} - 4\psi_{i,3} \pm 66hU_{tan}}{18h^2} + O(h^4) \tag{2.33}$$

Other relations were proposed by Woods (1954) and Orszag and Israeli (1974). It is very important to know that, the vorticity boundary condition is responsible to enforce no-slip boundary condition. Although non of them cannot fall velocity components to machine zero, but the accuracy in normal to the wall component of the velocity is two order of magnitude more than tangent component. The subject of the vorticity boundary condition has a long history, going back to Thom's formula in 1933 [1]. In a second order scheme, Thom's formula, Wilkes formula, or some other local formulas can be selected and coupled with a centered difference scheme at the interior points. The advantage of Thom's formula lies in its simplicity as only one interior point of the stream function is involved. Thus its stability is very robust. Yet, it was always very confusing why formulas like Thom's, which seems hopelessly first order by formal Taylor expansion on the boundary, is actually second order accurate. This mystery can be explained by Strang-type high-order expansions [19]. It was proven in 1964 that for nonlinear hyperbolic or parabolic equations, the L_2-norm stability for the linearized problem and the smoothness of the exact solution implies that the scheme has full accuracy (e.g., second order) in L_∞-norm. The main idea in the proof is the construction of high order expansions with respect to the scheme. Meth in his thesis [10], proved the stability for the linearized problem. The theoretical convergence analysis of the second order scheme with Thom's formula on the boundary was given by Hou and Wetton in [48]. It relied upon Strang-type high order expansions, which resulted in much more regularity assumption of the exact solution than needed. We should mention that the stability of Thom's formula cannot be applied to long-stencil formulas automatically, as Wilkes (1963) & Pearson's (1965) formula and other local formulas. It was a doubtful question for a long time: are these long-stencil formulas stable? This question is answered in [31] by performing a simple, clean analysis of the second order scheme using Wilkes formula to determine the vorticity at the boundary. In fact, direct calculations and standard local estimates cannot work it out. The convergence analysis of the compact scheme with Briley's [4] formula is given in [31], illustrating why the fourth order scheme combined with the third order boundary formula still works. Generally one can say that the order of implementation of the Neumann boundary conditions may be one order less than that of the discretization in the interior points of the solution domain. This choice

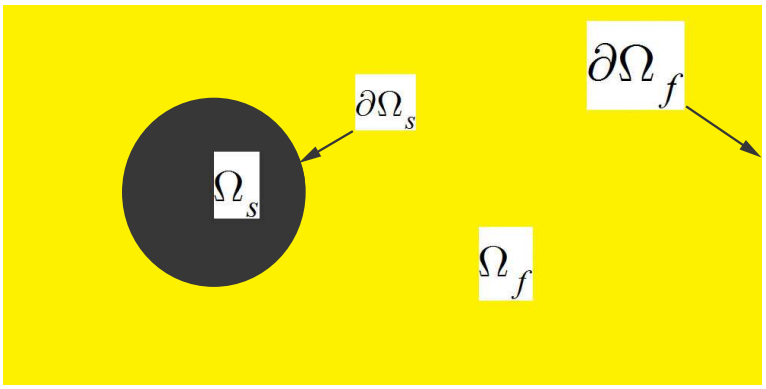


Figure 2.4: Domain of the solution and the immersed body, $\Omega = \Omega_f \cup \Omega_s$.

can keep the global desired accuracy of the discretization method. In the present investigation, Thom's formula at the no-slip boundaries is used for updating the vorticity. For more discussions, on boundary conditions for incompressible Navier-Stokes problems see [38].

2.6.1 Volume penalization method

For the simulation of curved solid boundaries which do usually not coincide with grid points one can use the volume penalization method proposed by Arquis and Caltagirone [7]. It is based on the idea of modeling solid bodies as porous media, thus getting ride of the Dirichlet boundary conditions by considering both the fluid and the solid part as one domain with different permeabilities, so one has a domain in which the solid is embedded. This method will lead to first-order accuracy near the solid boundary. In the Navier-Stokes equations (2.23) in primitive variables or the vorticity transport equation (2.26), penalization term can be added as a force term and thus, it is possible to introduce a solid body in the flow field. The penalization term for unit mass of the fluid reads,

$$\mathbf{F} = -\frac{\chi}{\eta}(\mathbf{u} - \mathbf{u}_B) \tag{2.34}$$

where $\mathbf{u}_B$ is the velocity of the immersed body (obstacle) which will be zero for fixed bodies. Penalization parameter η is the porosity (permeability) coefficient of the immersed body. In an explicit time integration Δt must be smaller than η. Typically we use $\eta = 10^{-3}$ or 10^{-4} to ensure the stability of time integration. The mask function χ describes the geometry of the flow domain, see Fig. 2.4

$$\chi(\mathbf{x}, t) = \begin{cases} 1 & \mathbf{x} \in \Omega_s \\ 0 & \mathbf{x} \in \Omega_f \end{cases} \tag{2.35}$$

where Ω_f represents the domain of the flow and Ω_s represents the immersed solid in the domain of the solution. The domain is governed by Navier-Stokes equations in the fluid regions and by Darcy's law in the solid regions, when $\eta \to 0$. With the use of the penalization method the hydrodynamic forces and the moments acting on

the obstacle, which are usually evaluated via surface integrals of the stress tensor $\sigma(\mathbf{u}, p) = \mu(\nabla\mathbf{u} + (\nabla\mathbf{u})^T)/2 - p\,\mathbf{I}$, can computed readily by integrating the penalized velocity over the obstacle's volume, thus we have the forces by

$$\mathbf{F} = \oint_{\partial\Omega_s} \sigma \cdot \mathbf{n}\,\mathrm{d}l = \lim_{\eta\to 0} \frac{\rho_f}{\eta} \int_{\Omega_s} \chi(\mathbf{u} - \mathbf{u}_B)\,\mathrm{d}s + \rho_f S_{pen} \ddot{\mathbf{x}}_{cg} \tag{2.36}$$

and moment in two-dimensions is evaluated by

$$M_{\mathbf{x}_{cg}} = \oint_{\partial\Omega_s} (\mathbf{x} - \mathbf{x}_{cg}) \times \sigma \cdot \mathbf{n}\,\mathrm{d}l = \lim_{\eta\to 0} \frac{\rho_f}{\eta} \int_{\Omega_s} \chi(\mathbf{x} - \mathbf{x}_{cg}) \times (\mathbf{u} - \mathbf{u}_B)\,\mathrm{d}s + \frac{\rho_f}{\rho_s} J_{cg} \ddot{\theta}_{cg} \tag{2.37}$$

in $[N.m]$, where J_{cg} is the moment of inertia taken about center of the mass, $\mathbf{n}$ is the unit outward vector to $\partial\Omega_s$, $\mathbf{x}_{cg}$ is the location of the center of gravity of the immersed body, θ is the angle of rotation around the center of gravity (dots denote derivation with respect to time) and S_{pen} is equivalent to the surface of penalized area

$$S_{pen} = \int_{\Omega_s} \chi\,\mathrm{d}s$$

For a uniform grid in two-dimensions numerically $\mathbf{F}$ can be evaluated as

$$\mathbf{F} \approx \frac{\rho_f}{\eta} \Delta x \Delta y \sum_{i=1}^{Imax} \sum_{j=1}^{Jmax} \chi_{i,j} (\mathbf{u} - \mathbf{u}_B)_{i,j} + \rho_f S_{pen} \ddot{\mathbf{x}}_{cg}^{n-1} \tag{2.38}$$

Finally the RHS of the vorticity equation including penalization term reads

$$RHS = -\frac{\partial\psi}{\partial y}\frac{\partial\omega}{\partial x} + \frac{\partial\psi}{\partial x}\frac{\partial\omega}{\partial y} + \nu\Big(\frac{\partial^2\omega}{\partial x^2} + \frac{\partial^2\omega}{\partial y^2}\Big) + \nabla \times \mathbf{F} \tag{2.39}$$

with the penalization term

$$\nabla \times \mathbf{F} = \frac{\partial}{\partial y}\Big(\frac{\chi}{\eta}\frac{\partial\psi}{\partial y}\Big) + \frac{\partial}{\partial x}\Big(\frac{\chi}{\eta}\frac{\partial\psi}{\partial x}\Big) - \frac{\partial}{\partial y}\Big(\frac{\chi u_B}{\eta}\Big) + \frac{\partial}{\partial x}\Big(\frac{\chi v_B}{\eta}\Big) \tag{2.40}$$

For the points for which the complete stencil belongs to the fluid domain ($\chi = 0$) we have $\nabla \times \mathbf{F} = 0$, but for the points for which the complete stencil belongs to the solid domain ($\chi = 1$) we have $\nabla \times \mathbf{F} = (\omega_B - \omega)/\eta$, i.e., in the time integration we will have $\omega^{n+1} \approx \omega_B = 2\Omega_B$, where Ω_B is the angular velocity of the embedded body. The penalization term is thus responsible for the vorticity production at the boundaries.

Chapter 3

Applications

3.1 Introduction

In this chapter the results obtained by applying the developed adaptive multiresolution method to the one-dimensional viscous Burgers equation and the two-dimensional incompressible Navier-Stokes equations will be presented. A dipole colliding with a straight no-slip wall for a Reynolds of 1000 will be considered as a benchmark, then with the use of the penalization method the collision of dipoles with curved walls, either concave or convex, will be examined for Reynolds= 10000.

3.2 Application to the one-dimensional Burgers equation

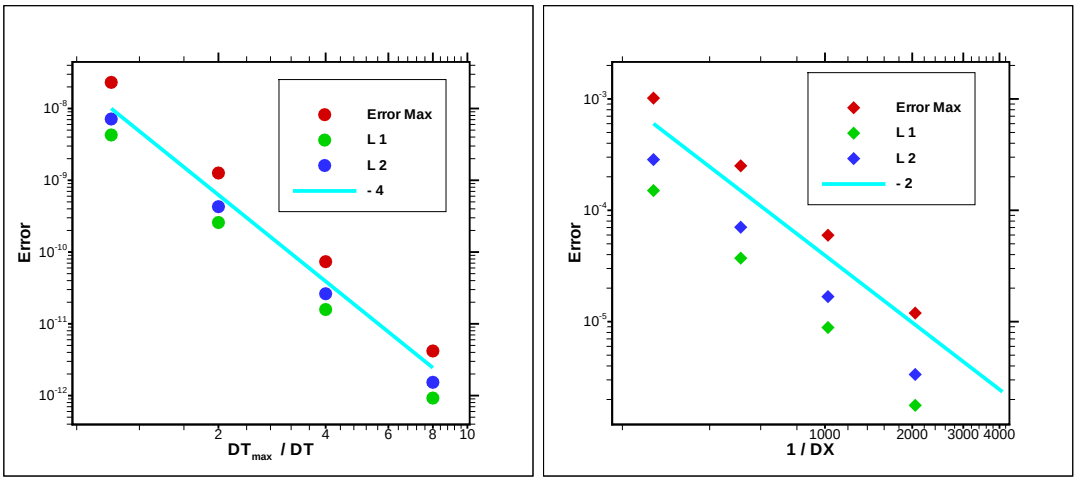
A well-known example of nonlinear PDEs is the one-dimensional viscous Burgers equation, including both nonlinear advection and diffusion terms.

$$\partial_t u + u u_x = \nu u_{xx} \tag{3.1}$$

The value of ν determines the ratio of two phenomena in competition. The Eq. (3.1) is completed with suitable initial and boundary conditions. By putting the spatial terms in RHS, using second-order centered finite differences for spatial discretization, the RHS operator reads

$$RHS(i) = -u_i \frac{u_{i+d} - u_{i-d}}{2d\Delta x} + \frac{u_{i+d} - 2u_i + u_{i-d}}{d^2 \Delta x^2} \tag{3.2}$$

where for a uniform grid d will be replaced by $d = 1$. First a solver based on uniform discretization of the Burgers equation via finite differences using Eq. (3.2) is developed as reference solver. An error analysis of the time integration of the Burgers equation (3.1) over a uniform grid is performed to examine the accuracy of the developed code based on the fourth order Runge-Kutta method. The aim is to show the decrease of the error with successive reduction of the time step. A simulation with $\Delta t_{max}/16$ is considered as reference solution to compute the errors, where Δt_{max} is chosen rather



(a) Error in time for the RK4 method. (b) Error in space for the 2^{nd} order method.

Figure 3.1: Time and space accuracy for the Burgers equation over uniform grid with $\nu = 10^{-3}$.

large, e.g., equal to 0.1, to avoid the truncation error from falling in the range of round-off error for $\Delta t_{max}/16$. The other simulations are performed with $\Delta t_{max}/8$, $\Delta t_{max}/4$, $\Delta t_{max}/2$ and Δt_{max}. The computations start from an initial condition

$$u(x,0) = \sin(x), \quad x \in [0, 2\pi]$$

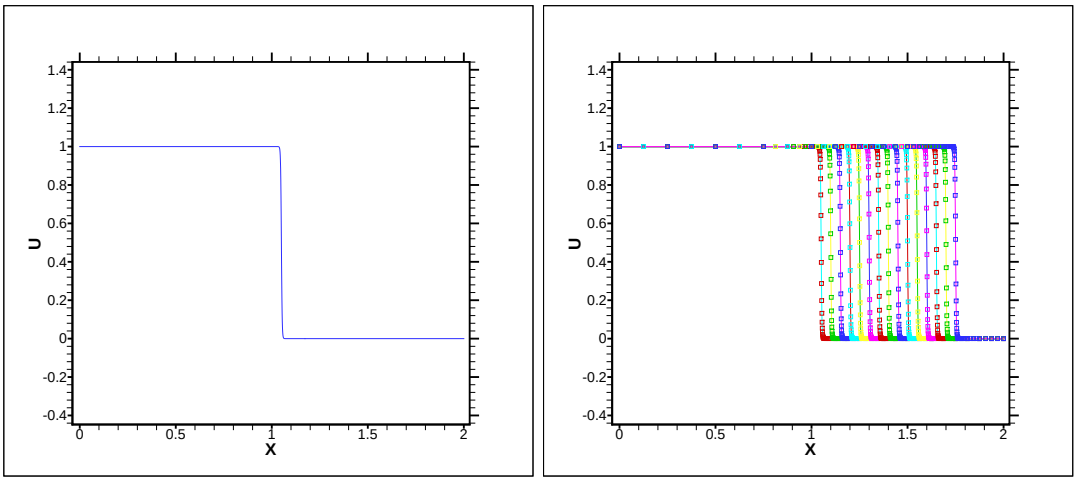
at $t = 0$ and stop at $t = 1$, so the time step and the number of iterations for each simulation is different (boundary conditions are fixed to zero at $x = 0, 2\pi$). Different errors as a function of $\Delta t_{max}/\Delta t$ are compared with the theoretical (-4) slope in Fig. 3.1 (a), a good agreement can be seen. Then an error analysis to examine the second-order discretization in space is performed. Again we consider the Burgers equation to show the accuracy of the solver on the uniform grid. A simulation with $J = 12$ grid points is considered as reference solution to compute the errors, other simulations were performed with 2^{11}, 2^{10}, 2^9 and 2^8 grid points. The computations start from an initial condition

$$u(x,0) = \sin(x), \quad x \in [0, 2\pi]$$

at $t = 0$ and stop at $t = 1$ (with fixed boundary conditions). The time step is chosen small enough, i.e., 4×10^{-5}, which is stable for time integration with 2^{12} grid points, to minimize the error due to time integration, consequently the number of iterations for all simulations are the same. Different errors are compared with the theoretical (-2) slope in Fig. 3.1 (b).

Then a one-dimensional adaptive solver based on the multiresolution algorithm presented in Chapter 2 and Eq. (3.2) for the viscous Burgers equation (3.1) is developed. Then we perform multiresolution computations for the viscous Burgers equation (3.1), for which analytical solutions are known. For $(x,t) \in \mathbb{R} \times [0, +\infty)$, an analytical solution in an infinite domain is given in the form of

$$u_{\text{exact}}(x,t) = \frac{1}{2}\left[1 - \tanh\left(\frac{x - 1 - t/2}{4\nu}\right)\right] \tag{3.3}$$



(a) Initial condition (b) Time evolution up to $t = 1.5$

Figure 3.2: Initial condition (left) and time evolution (right) computed with the multiresolution solver, threshold $\epsilon = 10^{-3}$, $\nu = 10^{-3}$, $\Delta t = 10^{-5}$ and a maximum grid level $J = 11$. The exact solution is represented by (—) solid color lines.

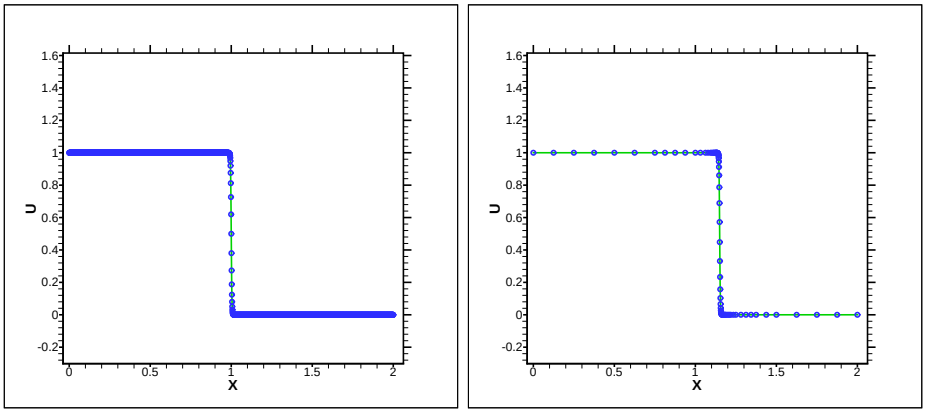
For $x \in [0, 2]$ it can be verified that the boundaries are sufficiently far so that their influence are negligible [33] for small time intervals. The initial condition is given by

$$u(x, 0) = \begin{cases} 1 & x \leq 1 \\ 0 & x > 1 \end{cases} \tag{3.4}$$

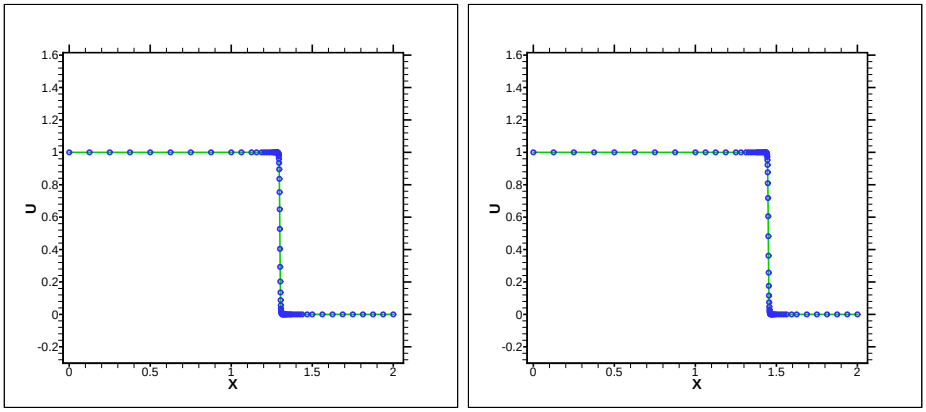
and Dirichlet boundary conditions at the right and left boundaries are imposed, i.e.,

$$u(0, t) = 1 \quad , \quad u(2, t) = 0.$$

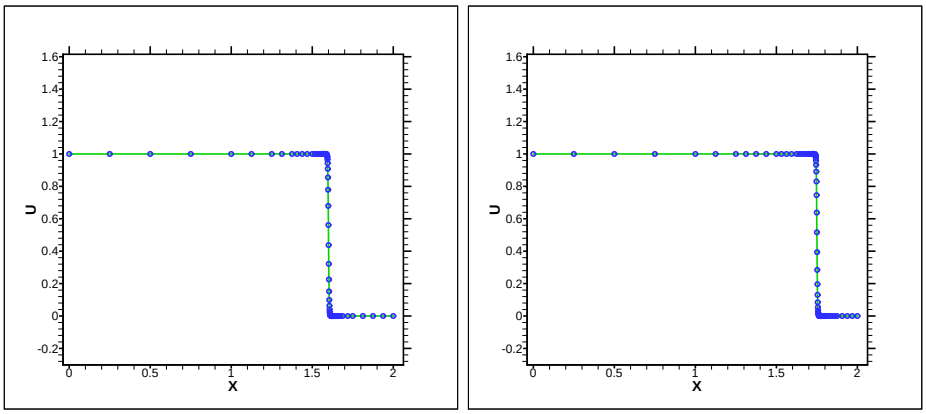
The initial condition and the time evolution of the solution computed with the multiresolution solver for Burgers equation, with threshold $\epsilon = 10^{-3}$ for $\nu = 10^{-3}$ and $\Delta t = 10^{-5}$, maximum grid level $J = 11$, are illustrated in Fig. 3.2 (a) and (b), respectively. The exact solution is plotted simultaneously by solid lines. The L_∞, L_1 and L_2 errors at $t = 1.5$ are of order 10^{-2}, 10^{-4} and 10^{-5}, respectively. The evolution of the active grid points during the computation, is illustrated in Fig. 3.3 and corresponding active points represented in their level, is illustrated in Fig. 3.4.
With the use of multiresolution code for the Burgers equation (3.1), some qualitatively comparisons of the results with uniform grid solver was performed. The only adjustable parameter which had a considerable impact in the agreement of the results with that of uniform grid solver is the threshold parameter (ϵ). Another error analysis is performed to examine the second-order accuracy of the multiresolution method in space by changing the threshold parameter. To show the conservation of the formal accuracy for the space discretization with adaptive method, the analytical exact solution in Eq. (3.3), is used as a reference solution to compute the errors, different simulations are performed with 2^{11}, 2^{10}, 2^9, 2^8 and 2^7 grid points. The computations start from the initial condition showed in Fig. 3.2 (a) at $t = 0$ and stop at $t = 1.5$ showed in Fig. 3.3. The time step is chosen smaller than stability limit for 2^{11} grid



(a) $t = 0$ (b) $t = 0.3$

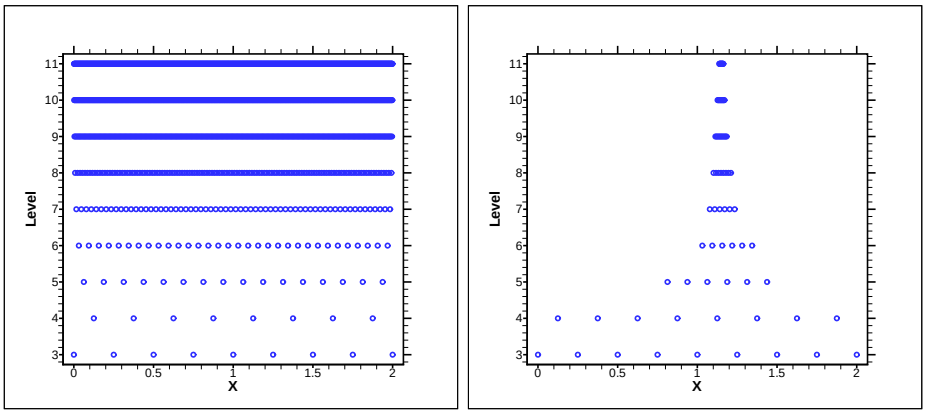


(c) $t = 0.6$ (d) $t = 0.9$

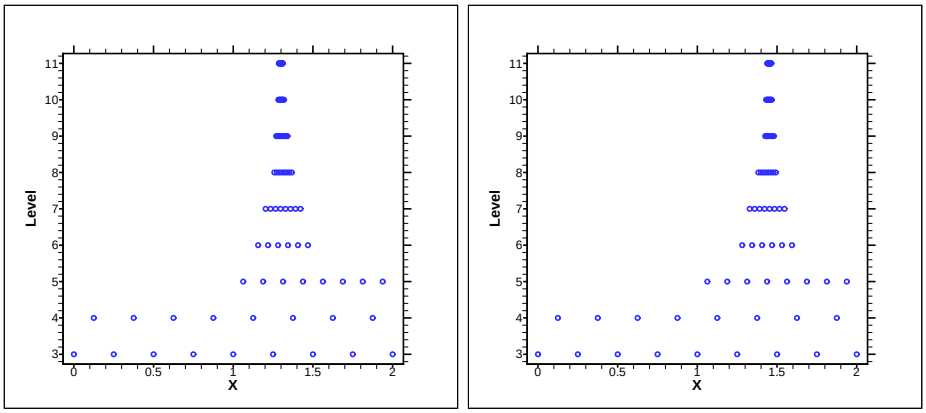


(e) $t = 1.2$ (f) $t = 1.5$

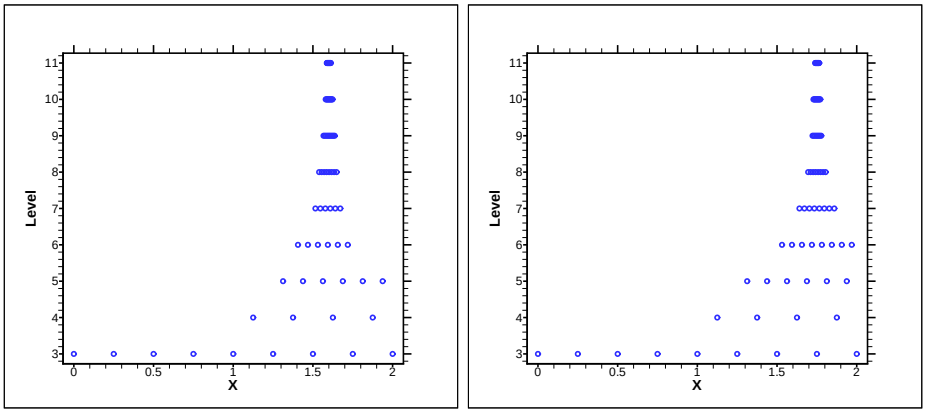
Figure 3.3: The evolution of the adaptive solution during time (active points represented by ○) computed with the multiresolution solver and the exact solution given by Eq. (3.3) for the Burgers equation, $\nu = 10^{-3}$, (represented by ——), with maximum grid level $J = 11$ and a threshold parameter $\epsilon = 10^{-3}$.



(a) $t = 0$ (b) $t = 0.3$

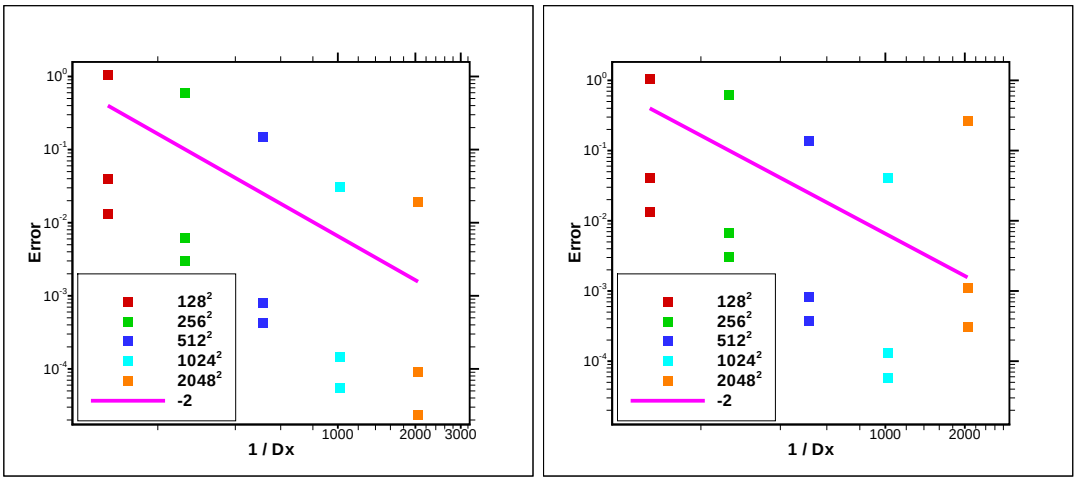


(c) $t = 0.6$ (d) $t = 0.9$



(e) $t = 1.2$ (f) $t = 1.5$

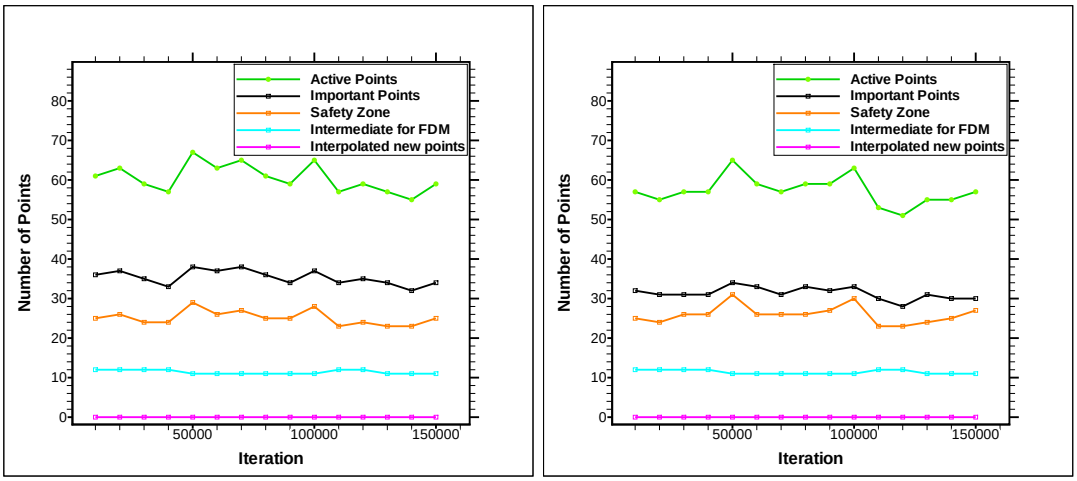
Figure 3.4: The evolution of the active grid points during time represented in their level (marked by ○) during the computation with the multiresolution solver for the Burgers equation, $\nu = 10^{-3}$, with maximum grid level $J = 11$, minimum level of filtering $j = 3$, threshold parameter $\epsilon = 10^{-3}$.



(a) $\epsilon = 10^{-3}$ (b) $\epsilon = 5 \times 10^{-3}$

Figure 3.5: Convergence of different spatial errors (L_∞ on top, L_1 on bottom, for all cases) for the Burgers equation computed with the multiresolution solver, for $\nu = 10^{-3}$ and $\Delta t = 10^{-5}$.

points, to minimize the error due to time integration. Hence the number of iterations for all simulations are the same. Different errors are compared with the theoretical (-2) slope in Fig. 3.5 (a) and (b) for $\epsilon = 10^{-3}$ and $\epsilon = 5 \times 10^{-3}$, respectively. The results are in good agreement with the (-2) slope, which confirms that the adaptive multiresolution method preserves the discretization order of the underlying numerical scheme. There is a deviation for $\epsilon = 5 \times 10^{-3}$, with 2^{11} grid points, which is due to big threshold parameter in comparison with truncation error. The truncation error of a second-order method in space $x \in [0, 2\pi]$ is of order 10^{-5} for 2^{11} grid points.
The evolution of the number of active, important (corresponding to the retained points after filtering of wavelet coefficients), safety zone, hung and interpolated points for wavelet transform during the computation with the multiresolution solver for the Burgers equation with $\nu = 10^{-3}$ and $\Delta t = 10^{-5}$ with maximum grid level $J = 10$, for threshold $\epsilon = 10^{-3}$ and $\epsilon = 5 \times 10^{-3}$, are illustrated in Fig. 3.6, (a) and (b), respectively. The average number of points after filtering is about 35 and after adding the safety zone, it increases to 63 out of 1025. Hence for this test case we have a compression of points about 95% which is considerable, and the time of computations with the multiresolution solver is at least half in comparison to the uniform grid solver. The gain in CPU-time becomes more especially for fine resolutions, where the overhead due to the additional complexity of the multiresolution algorithm becomes less important.



(a) $\epsilon = 10^{-3}$ (b) $\epsilon = 5 \times 10^{-3}$

Figure 3.6: The evolution of the number of active, important (corresponding to the retained points after filtering of wavelet coefficients), safety zone, hung and interpolated points for the wavelet transform during the computation with the multiresolution solver for the Burgers equation, with maximum grid level $J = 10$ (1025 grid points).

3.3 Application to two-dimensional incompressible flows

In this section we are considering the two-dimensional incompressible flows by means of the vorticity transport equation (2.26). By putting all of the spatial derivatives in the RHS operator, the discretized form of the vorticity equation including the penalization term (2.34) reads

$$\partial_t \omega|_{i,j} = RHS(i,j)$$

with the use of second-order finite-difference approximations (2.6), (2.7) and (2.8) one gets:

$$\begin{aligned} RHS(i,j) = &-\frac{\psi_{i,j+d}-\psi_{i,j-d}}{2d\Delta y}\frac{\omega_{i+d,j}-\omega_{i-d,j}}{2d\Delta x} + \frac{\psi_{i+d,j}-\psi_{i-d,j}}{2d\Delta x}\frac{\omega_{i,j+d}-\omega_{i,j-d}}{2d\Delta y} \\ &+\nu\left(\frac{\omega_{i+d,j}-2\omega_{i,j}+\omega_{i-d,j}}{d^2\Delta x^2} + \frac{\omega_{i,j+d}-2\omega_{i,j}+\omega_{i,j-d}}{d^2\Delta y^2}\right) \\ &+\frac{(\chi_{i+d,j}+\chi_{i,j})(\psi_{i+d,j}-\psi_{i,j}) - (\chi_{i-d,j}+\chi_{i,j})(\psi_{i,j}-\psi_{i-d,j})}{2\eta d^2\Delta x^2} \\ &+\frac{(\chi_{i,j+d}+\chi_{i,j})(\psi_{i,j+d}-\psi_{i,j}) - (\chi_{i,j-d}+\chi_{i,j})(\psi_{i,j}-\psi_{i,j-d})}{2\eta d^2\Delta y^2} \\ &-\frac{(u^B_{i,j+d}\chi_{i,j+d} - u^B_{i,j-d}\chi_{i,j-d})}{2\eta d\Delta y} + \frac{(v^B_{i+d,j}\chi_{i+d,j} - v^B_{i-d,j}\chi_{i-d,j})}{2\eta d\Delta x} \end{aligned}$$

For a uniform grid in the above formula d will be replaced by 1, for the multiresolution solver d will be determined by a search algorithm, to find the nearest active point.

The vorticity equation is advanced in time with a Runge-Kutta technique which was described in Section 2.4.1. Before each evaluation of RHS, one Poisson equation for stream function with vorticity (ω) forcing on the right hand side, i.e., Eq. (2.27) must be solved to update the stream function to the correct value for the interior points. The vorticity at the boundary points must be updated by Thom's formula. The Poisson equation will be solved via an iterative method for elliptic equations, namely the point successive over relaxation (PSOR) method

$$\psi_{i,j}^{n+1} = \beta\psi_{i,j}^{new} + (1-\beta)\psi_{i,j}^{old} \tag{3.5}$$

where β is the over relaxation factor, $\beta \in [1,2]$. So we will have

$$\psi_{i,j}^{n+1} = \psi_{i,j}^{old} + \frac{\beta}{4}Res \tag{3.6}$$

by the use of the Gauss-Seidel method

$$Res = \sum_{nb=1}^{4} \psi_{nb}^{last} - 4\psi_{i,j}^{old} + h^2\omega_{i,j} \tag{3.7}$$

in which $h = \Delta x = \Delta y$ and nb denotes the neighbor points with distance d (always their updated value will be used). The norm of the residuum, $||Res||_\infty$ must converge below to a prescribed convergence criterion, e.g., 10^{-6}. For the finer grids like $J = 12$ we require smaller criterion, e.g., 10^{-12}. For useful discussions in the choice of a proper over relaxation factor we refer to [18]. The hung points are the same for the elliptic and the parabolic part and they must be updated in each iteration. In the present investigation a geometric multi-grid solver for uniform grids will be used to accelerate the convergence of the elliptic part, which is the most time-consuming part of the simulations. But the PSOR solver is used in the multiresolution simulations because for the moment the MG solver is not applicable for adaptive grids. See [37] for a multilevel adaptive method for elliptic problems and [39] for a MG method used in adaptive solution of incompressible flows via FDM. In terms of memory usage the PSOR solver is more advantageous in comparison to MG solver, thus for the moment the applicability of multigrid solver due to insufficient memory is limited to maximum grid level $J = 10$ in each direction.

3.4 Dipole-wall collision

The famous problem of the dipole collision with a straight wall is considered in this section. We will study the flow in a square domain $[0,2] \times [0,2]$. On the four walls of the domain ($x, y = 0$ & $x, y = 2$) no-slip and no-penetration boundary conditions are applied. The flow is initialized in the form of two shielded Gaussian mono-polar vortices, their centers placed at a distance 0.2 apart. The vorticity distribution in each monopole is given by

$$\omega(0, \mathbf{x_n}) = \omega_e\Big(1 - \frac{\mathrm{r}^2}{\mathrm{r}_0^2}\Big)\exp\Big(-\frac{\mathrm{r}^2}{\mathrm{r}_0^2}\Big) \tag{3.8}$$

where r_0 is the core radius, $r = ||\mathbf{x} - \mathbf{x}_n||$ with $\mathbf{x}_n$ being the position of the vortex center. The two isolated monopoles are located at

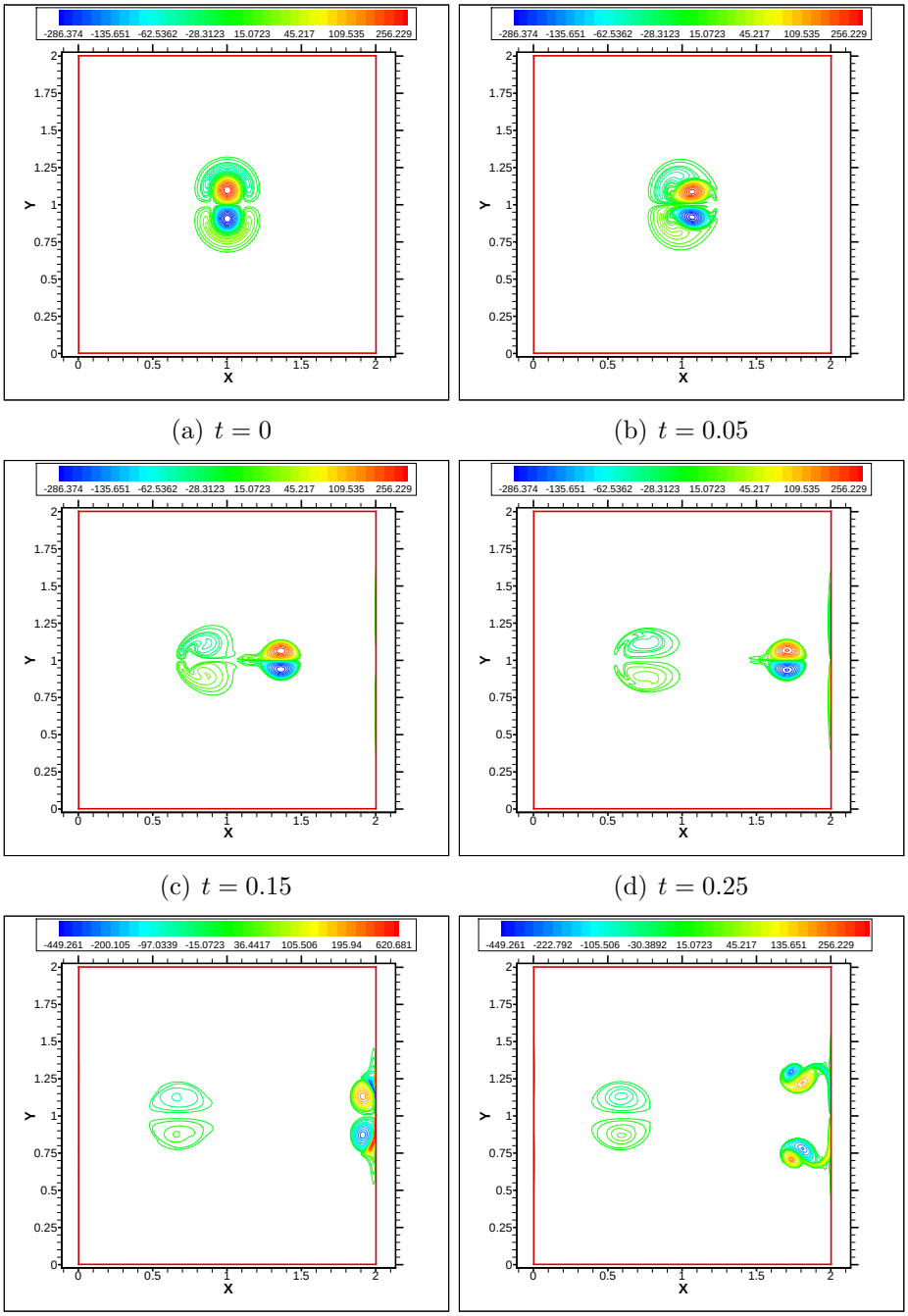
$$\mathbf{x}_1 = (1, 1.1) \text{ and } \mathbf{x}_2 = (1, 0.9)$$

Demanding that the root mean square (rms) velocity is initially equal to unity ($E = 2$) yields the amplitude of each isolated monopole, $\omega_e = \pm 299.528385375226$ [47]. The core radius of the shielded monopoles is set to $r_0 = 0.1$. The vorticity amplitude in the radial direction decreases exponentially with r. As a result, the circulation of one isolated monopole calculated over a circular contour around the vortex origin decreases exponentially towards zero for increasing contour radius. Hence, no boundary layers are required at the no-slip walls when constructing the initial flow field. The integral-scale Reynolds number for the initial field is given by

$$Re = \frac{U_{rms} L}{\nu} \tag{3.9}$$

where the characteristic length scale is set to the half-height of the domain, $L = 1$, and the characteristic velocity to the initial root mean square velocity, $U_{rms} = 1$. This integral Reynolds number differs slightly from the Reynolds number $Re_d \approx 0.8Re$ based on the dipole translation speed U_d and the dipole radius R, see [45]. The time evolution of the dipole is calculated by the developed multiresolution finite difference code with threshold $\epsilon = 10^{-3}$ and maximum grid level $J = 11$ for Reynolds 1000. The evolution of the vorticity isolines starting from the initial condition at $t = 0$ up to $t = 1$, is shown in Fig. 3.7 and Fig. 3.8. After releasing the initial set of shielded mono-polar vortices, Fig. 3.7 (a), one observes how the cores combine into a dipole, while the surrounding shields are substantially deformed Fig. 3.7 (b). Somewhat later, the dipolar vortex has traveled away in the positive x-direction Fig. 3.7 (c), and the shields left behind combine into another, much weaker dipolar structure that translates in the opposite direction Fig. 3.7 (d). This weak dipole has no considerable impact on the primary dipole and is not discussed further. The first collision with the right wall can be seen in Fig. 3.7 (e) and the formation of secondary vortices is illustrated in Fig. 3.7 (f). Previous studies have revealed that the primary dipole vortex closely resembles the Lamb dipole, i.e., a compact dipolar vorticity structure in a circular area with a linear vorticity stream-function relationship. The rest of the evolution and the collision of the primary and newly generated vortices with right and left walls are illustrated in Fig. 3.8; detachment of vortices from wall is seen in Fig. 3.8 (a), a second collision of newly generated dipoles with the right wall in Fig. 3.8 (b), formation and turnover of weaker secondary vortices in Fig. 3.8 (c), (d), (e), collision of the backward traveling dipole with the left wall in Fig. 3.8 (f).

The evolution of the adaptive grid during the computation of the dipole-wall collision with the multiresolution solver with threshold $\epsilon = 10^{-3}$ and maximum grid level $J = 11$, for Reynolds 1000 are illustrated in Fig. 3.9 and Fig. 3.10. The evolution of the number of active, safety zone, hung and interpolated points for the wavelet transform during the computation of the dipole-wall collision with the multiresolution solver with maximum grid level $J = 11$, Reynolds 1000, for two thresholds, $\epsilon = 10^{-3}$

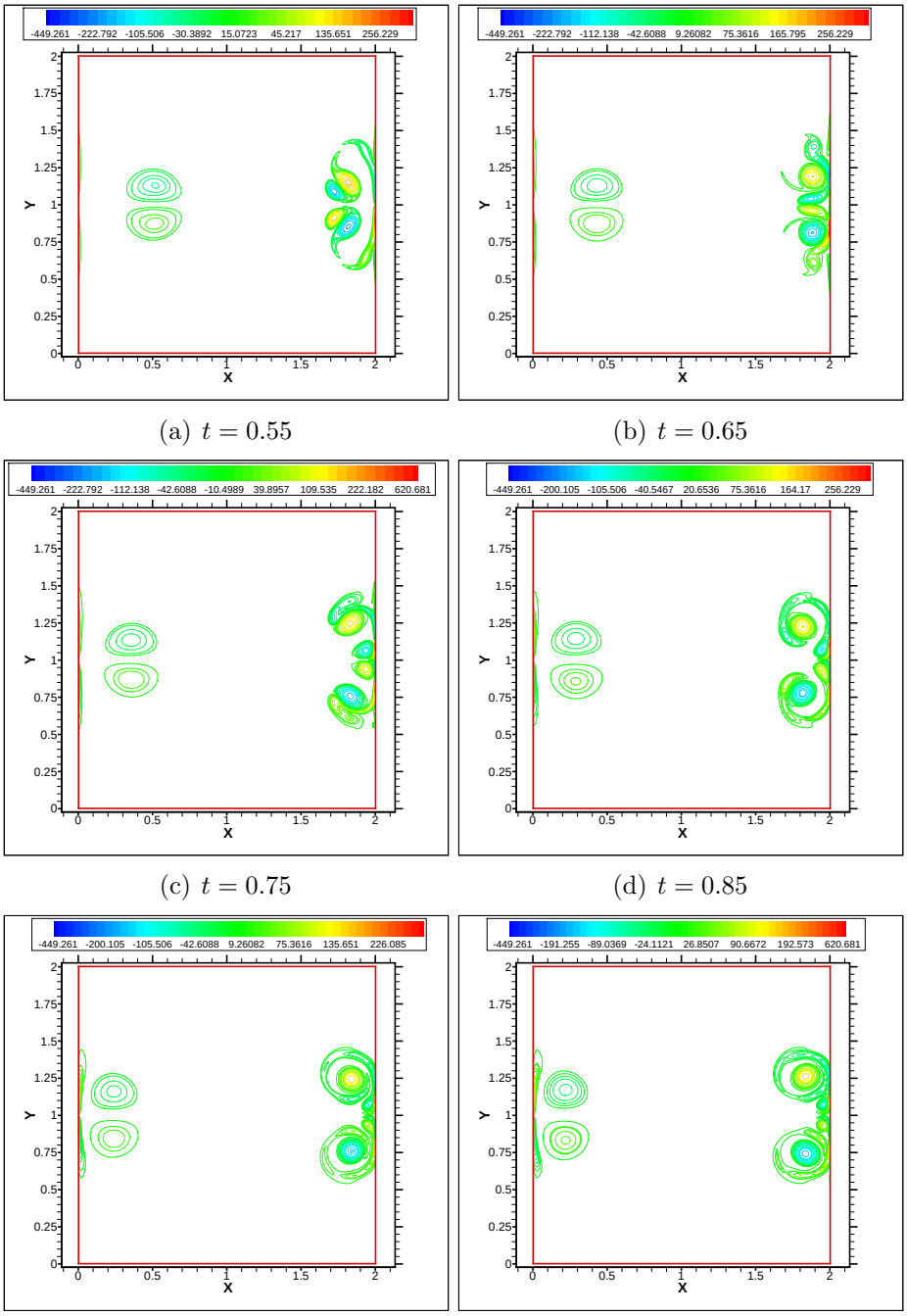


(a) $t = 0$ (b) $t = 0.05$

(c) $t = 0.15$ (d) $t = 0.25$

(e) $t = 0.35$ (f) $t = 0.45$

Figure 3.7: The evolution and collision of the vortices (represented by the colored isolines) with walls, maximum grid level $J = 11$ in each direction, threshold $\epsilon = 10^{-3}$, for Reynolds 1000. (a) The initial vorticity distribution; (b) Shedding of the surrounding vorticity shields; (c) Formation of a forward-traveling Lamb-type dipole; (d) Formation of a weaker backward-traveling dipole; (e) First collision with the right wall; (f) Formation of secondary vortices.

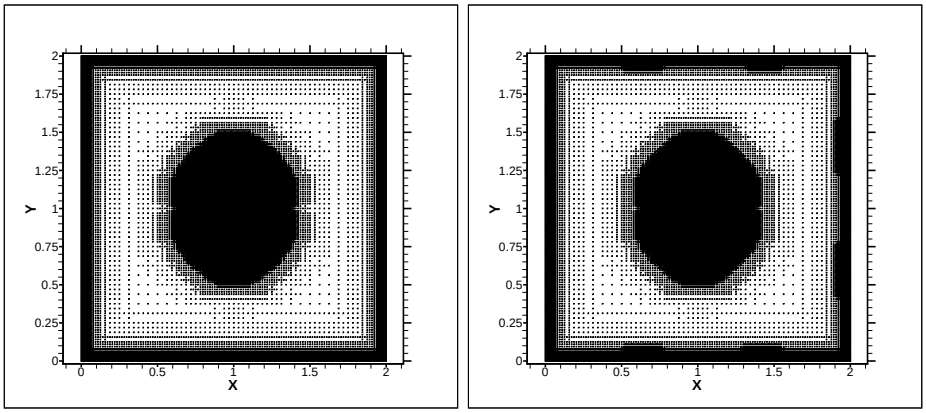


(a) $t = 0.55$ (b) $t = 0.65$

(c) $t = 0.75$ (d) $t = 0.85$

(e) $t = 0.95$ (f) $t = 1$

Figure 3.8: The evolution and collision of the vortices (represented by the colored isolines) with walls, maximum grid level $J = 11$ in each direction, threshold $\epsilon = 10^{-3}$, for Reynolds 1000. (a) Detachment of vortices from the wall; (b) Second collision of newly generated dipoles with the right wall; (c-d-e) Formation and turnover of weaker secondary vortices; (f) Collision of backward-traveling dipole with the left wall.

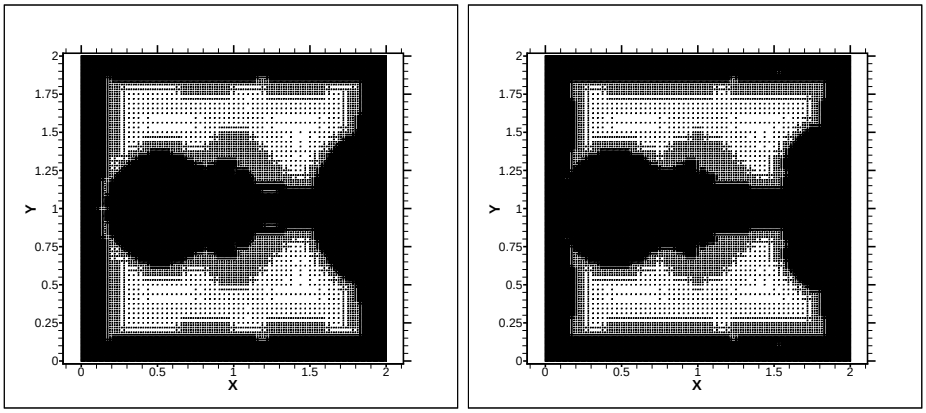


(a) $t = 0$ (b) $t = 0.5$

(c) $t = 0.15$ (d) $t = 0.25$

(e) $t = 0.35$ (f) $t = 0.45$

Figure 3.9: Evolution of the adaptive grid during the computation of the dipole-wall collision with the multiresolution solver with threshold $\epsilon = 10^{-3}$, maximum grid level $J = 11$ in each direction for Reynolds 1000.

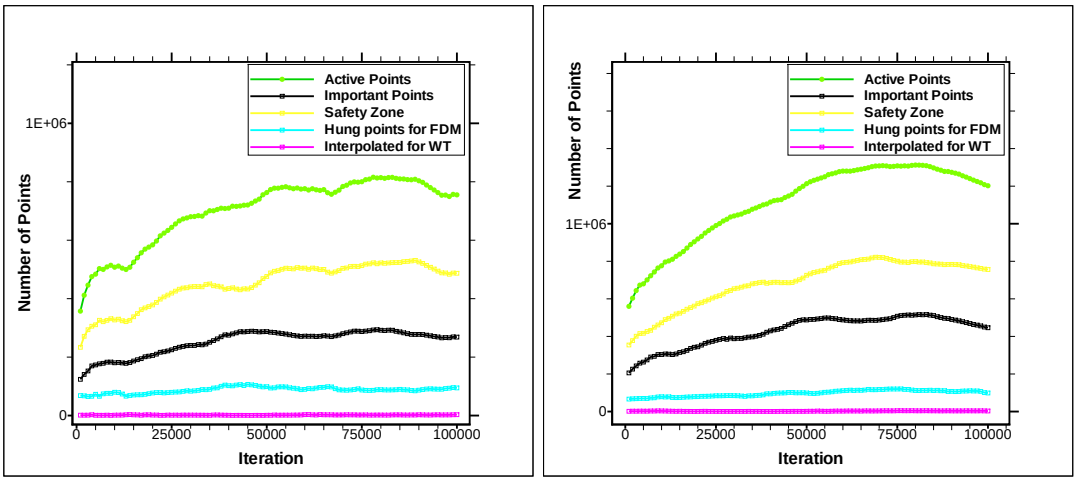


(a) $t = 0.55$ (b) $t = 0.65$

(c) $t = 0.75$ (d) $t = 0.85$

(e) $t = 0.95$ (f) $t = 1$

Figure 3.10: Evolution of the adaptive grid during the computation of the dipole-wall collision with the multiresolution solver with threshold $\epsilon = 10^{-3}$, maximum grid level $J = 11$ in each direction for Reynolds 1000.



(a) $\epsilon = 10^{-3}$ (b) $\epsilon = 10^{-4}$

Figure 3.11: Evolution of the number of active, safety zone, hung and interpolated points for the wavelet transform during the computation of the dipole-wall collision with the multiresolution solver, maximum grid level $J = 11$ in each direction.

and $\epsilon = 10^{-4}$, are illustrated in Fig. 3.11, (a) and (b), respectively. The average number of points after filtering is about 3×10^5 and after adding the safety zone it is increased to 10^6. Hence for this test case we have a compression of points about 25% which is not so good in comparison to one-dimensional test case, but the time of the computations with multiresolution solver is at least one over six in comparison to the uniform grid solver. This CPU-time is promising, especially for higher resolutions with large number of grid points, i.e., J larger than nine. The main reason for considerable CPU-reduction is the presence of the elliptic solver in the algorithm in which the order of operations is proportional to $O(N^2)$ for a point successive over relaxation method, where N is the number of active points.

3.4.1 Convergence study*

To verify the accuracy of the numerical method the results are compared with the results reported by Clercx et al. [42] which are computed with a pseudo spectral solver. The method of Clercx et al. is a pseudo spectral method, both the velocity and vorticity are expanded in a truncated series of Fourier polynomials for the periodic-direction and in a truncated series of Chebyshev polynomials for the non periodic-direction. Some invariants of the flow, (i.e. total energy, enstrophy and palinstrophy) which are conserved by the flow dynamics for inviscid fluids ($\nu = 0$), can be assessed in viscous flows, where they will not conserved, but instead varying in time depending on the Reynolds number. Three quantities in the flow field, i.e., total energy E, total

enstrophy Z and total palinstrophy*[1] P are defined as

$$E(t) = \frac{1}{2}\int_{\Omega} |\mathbf{u}(\mathbf{x},t)|^2 d\mathbf{x} \tag{3.10}$$

$$Z(t) = \frac{1}{2}\int_{\Omega} |\omega(\mathbf{x},t)|^2 d\mathbf{x} \tag{3.11}$$

$$* \; P(t) = \frac{1}{2}\int_{\Omega} |\nabla\omega(\mathbf{x},t)|^2 d\mathbf{x} \tag{3.12}$$

Applying the trapezoidal quadrature formula we get the discrete versions

$$E(t) = \frac{\Delta x \Delta y}{2} \sum_{i=1}^{Imax} \sum_{j=1}^{Jmax} (u_{i,j}^2 + v_{i,j}^2) \tag{3.13}$$

$$Z(t) = \frac{\Delta x \Delta y}{2} \sum_{i=1}^{Imax} \sum_{j=1}^{Jmax} (\omega_{i,j})^2 \tag{3.14}$$

$$* \; P(t) = \frac{\Delta x \Delta y}{2} \sum_{i=1}^{Imax} \sum_{j=1}^{Jmax} \left(\frac{\partial \omega_{i,j}}{\partial x}\right)^2 + \left(\frac{\partial \omega_{i,j}}{\partial y}\right)^2 \tag{3.15}$$

where $\Delta x = \frac{L_x}{Imax-1}$ and $\Delta y = \frac{L_y}{Jmax-1}$. For any two-dimensional viscous flow ($z > 0$) the total energy $E(t)$ decays according to

$$\frac{dE}{dt} = -\nu \int_{\Omega} \omega^2 dA = -2\nu Z. \tag{3.16}$$

where $-2\nu Z$ is the energy dissipation [6]. Note that the decay rate is proportional to the total enstrophy, Z(t), which is a measure of the squared vorticity integrated over the domain. Understanding the evolution of the total enstrophy is therefore of crucial importance for explaining the energy decay. For a domain with no-slip boundaries the change in total enstrophy is governed by

$$\frac{dZ}{dt} = -2\nu P + \nu \oint_{\partial\Omega} \omega(\mathbf{n} \cdot \nabla\omega) ds \tag{3.17}$$

where $\mathbf{n}$ denotes the outer normal vector with respect to $\partial\Omega$. The first term on the right-hand side simply states that the enstrophy decays due to vorticity gradients (palinstrophy) that are present in the flow. The second term represents the vorticity production at the no-slip boundaries involving the vorticity and its gradients, which will give rise to the total palinstrophy. Note that the vorticity influx at the no-slip boundaries is equal to $(\mathbf{n} \cdot \nabla\omega)$. In the case of a square domain with stress-free or

[1]*Notice: There is an error in some curves of palinstrophy $P(t)$ noted by *, because of not taking to account the vorticity derivatives at boundaries via forward/backward stencil, which are significant. Otherwise all the curves of palinstrophy $P(t)$ obtained by present study will be above that of Clerx et al. as is in the case of enstrophy. I am apologizing the reader to not be able to modify the figures now but in a probable future publication this will be considered.

periodic boundary conditions the second term on the right-hand side of Eq. (3.17) vanishes. As a result, the total enstrophy cannot increase for a domain with stress-free or periodic boundary conditions and is thus always bounded by its initial value and zero [45]. For a steady flow we have

$$P = \frac{1}{2} \oint_{\partial\Omega} \omega(\mathbf{n} \cdot \nabla\omega) ds$$

Different simulations were performed to obtain a grid independent solution, by successive increasing the number of points. To have a stable simulation the time step must be reduced according to the CFL condition, see Table 3.1. A simulation over an uniform grid with a second order (in space) multi-grid solver with maximum grid level $J = 10$ in each direction, was done in the first step. The evolution of the total kinetic energy, total enstrophy and total palinstrophy for $Re = 1000$ are compared with the computations of Clercx et al. [42] in Fig. 3.12 and Fig. 3.13 (a) and (b), respectively. Note that the energy steadily decreases from its normalized initial value of $E = 2$ towards $E \approx 0.8$ at $t = 1$. At $t \approx 0.35$ the first collision of the dipole with the right wall takes place and the kinetic energy decays faster, which is due to the increased enstrophy production (dissipation) on the domain. On the other hand enstrophy starts from its initial value $P(0) = 800$ and decays until $t = 0.2$ and then goes up rapidly. The first peak in the enstrophy curve during time takes places at $t = 0.35$, and thus coincides with the first collision of the dipole with the right wall. During the first collision the boundary layers create a large amount of vorticity. The enstrophy in the boundary layers is then the main contribution to the total enstrophy. At $t \approx 0.64$ another smaller peak is visible in the enstrophy evolution curve, which is due to the second collision of newly generated vortices. The results for total energy are in reasonable agreement with the pseudo-spectral simulations, we can see a systematic deviation in enstrophy and palinstrophy curves especially near the first peak, which is due to insufficient resolution of second-order solver in comparison with the pseudo-spectral solver of Clercx [29].

Two simulations for Reynolds 1000 with the multiresolution code, using a maximum grid level $J = 11$, were performed with $\epsilon = 10^{-3}$ and $\epsilon = 10^{-4}$. The total enstrophy and palinstrophy during time, with $\epsilon = 10^{-3}$, are compared with the computations of Clercx et al. in Fig. 3.14, (a) and (b), respectively. The same quantities with $\epsilon = 10^{-4}$ are compared with that of Clercx et al. in Fig. 3.15, (a) and (b), respectively. The accordance of enstrophy and palinstrophy curves during time with Clercx data with $J = 11$ are better than that of $J = 10$, which is promising. Another point to be noted is that the results for $\epsilon = 10^{-4}$ and $\epsilon = 10^{-3}$ are almost identical and the multiresolution code works properly with $\epsilon = 10^{-3}$. Hence we will use $\epsilon = 10^{-3}$ for the multiresolution computations. The number of grid points and the corresponding time steps and CPU times on an Intel-P4 CPU (1.3 GHz), are given in the Table 3.1. In one Runge-Kutta iteration (averaged over 100 Δt) four Poisson equation (2.27) must be solved using either multigrid or PSOR methods.

A convergence study for the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ (with the uniform grid solver) for Reynolds 1000, with different grid spacings, i.e., maximum level in each direction $J = 8, 9, 10, 11$, is performed. Once again the sim-

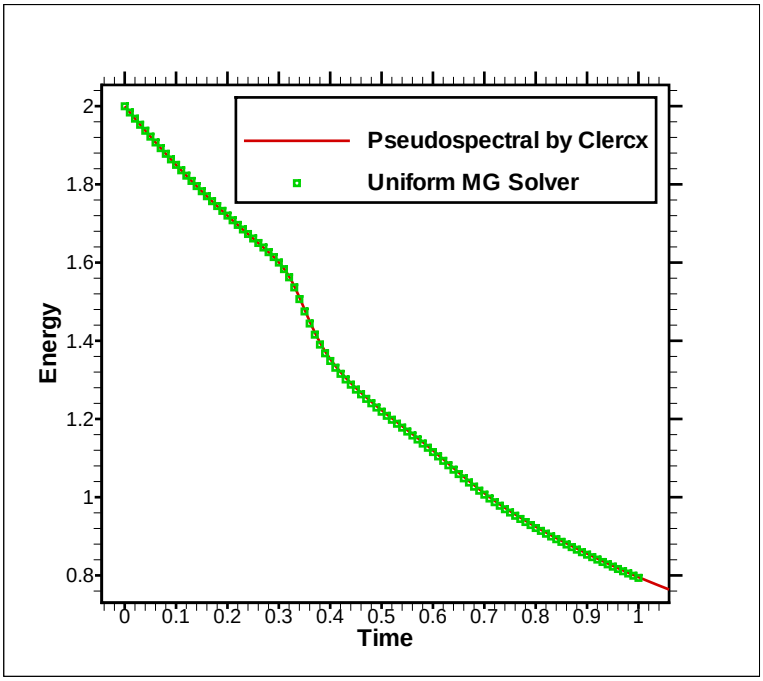
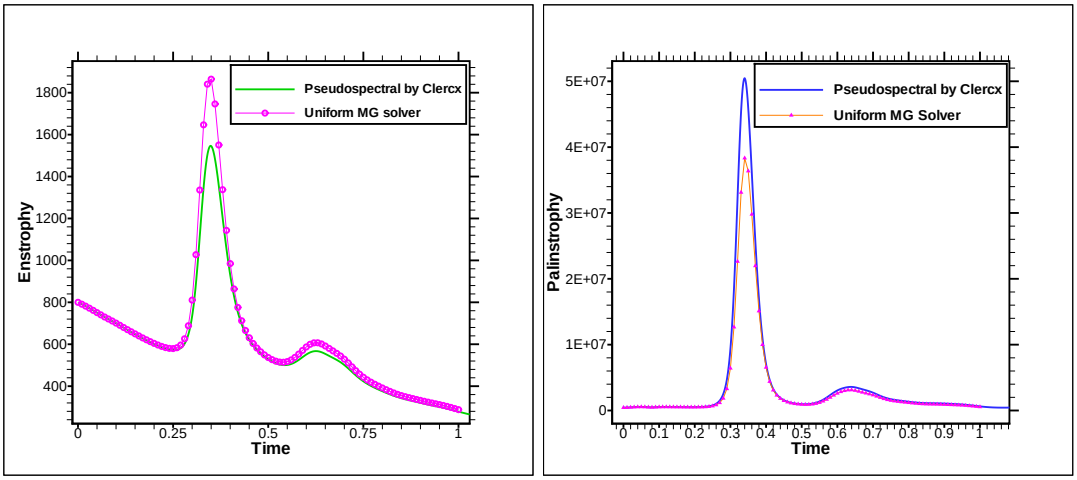


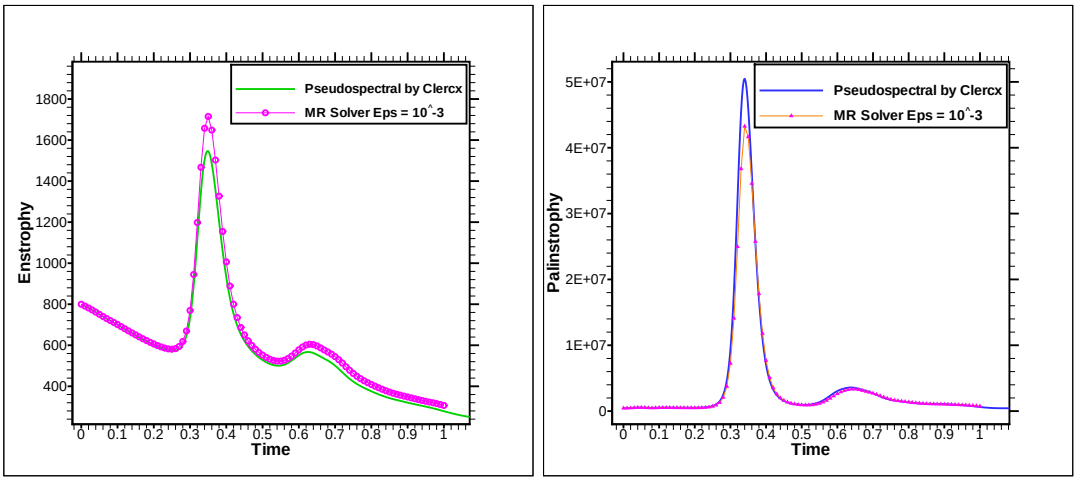
Figure 3.12: Comparison of the total energy $E(t)$ between the data from Clercx and the present finite difference computation with a uniform multigrid solver for Reynolds 1000 and maximum grid level $J = 10$ in each direction.



(a) Total enstrophy

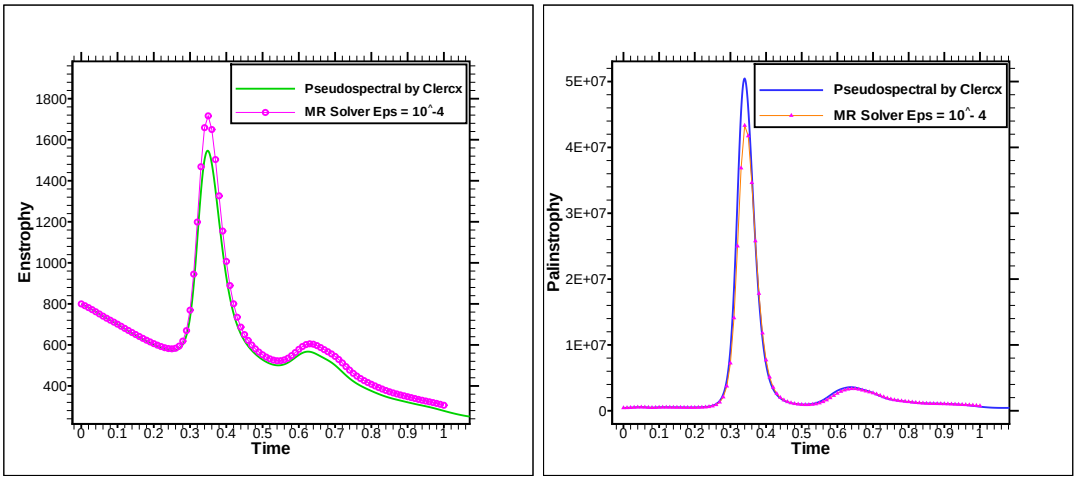
(b) Total palinstrophy*

Figure 3.13: Comparison of the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ between the data from Clercx and the present finite difference computation with a uniform multigrid solver for Reynolds 1000 and maximum grid level $J = 10$ in each direction.



(a) Total enstrophy (b) Total palinstrophy*

Figure 3.14: Comparison of the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ between the data from Clercx and the present computation with a multiresolution solver with threshold $\epsilon = 10^{-3}$ for Reynolds 1000 and maximum grid level $J = 11$ in each direction.



(a) Total enstrophy (b) Total palinstrophy*

Figure 3.15: Comparison of the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ between the data from Clercx and the present computation with a multiresolution solver with threshold $\epsilon = 10^{-4}$ for Reynolds 1000 and maximum grid level $J = 11$ in each direction.

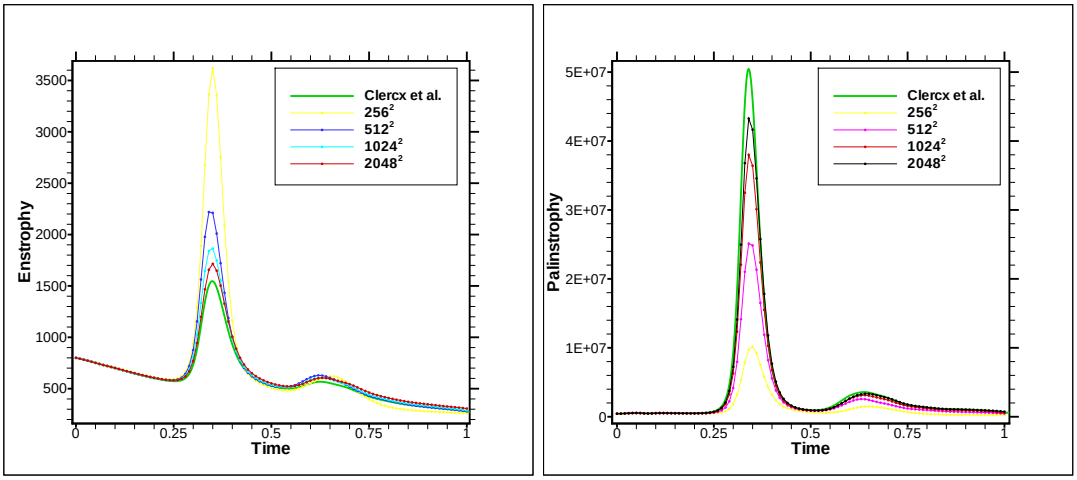
Table 3.1: Number of grid points and the corresponding time steps Δt and CPU time (second) for one RK4 iteration with multigrid and PSOR solver with $||Res||_\infty \leq 10^{-7}$, for $Re = 1000$ and maximum grid level J in each direction.

Grid	J	Δt	MG	PSOR
65×65	6	1×10^{-2}	0.003	0.01
129×129	7	5×10^{-3}	0.01	0.06
257×257	8	1×10^{-3}	0.07	0.2
513×513	9	5×10^{-4}	0.6	1.2
1025×1025	10	1×10^{-4}	2.5	7
2049×2049	11	5×10^{-5}	—	74
4097×4097	12	1×10^{-5}	—	—

ulation with pseudo-spectral solver of Clercx is taken as the reference solution. The results of the present computation are illustrated in Fig. 3.16, (a) and (b), respectively. It can be observed that by increasing the number of grid points the curves become closer and closer, we hope the results of $J = 12$ will match with that of Clercx. Now comparisons of total energy $E(t)$, total enstrophy $Z(t)$ and total palinstrophy $P(t)$ between the uniform grid solver and the multiresolution computation with thresholds, $\epsilon = 10^{-3}$ and $\epsilon = 10^{-4}$, for Reynolds 1000 and with maximum grid level $J = 9$ will be presented. The comparison of total energy $E(t)$ is plotted in Fig. 3.17, the results for total enstrophy $Z(t)$ and total palinstrophy $P(t)$ are compared in Fig. 3.18, (a) and (b), respectively. The agreement between the uniform grid solver and the multiresolution solver is perfect and the results for $\epsilon = 10^{-3}$ and $\epsilon = 10^{-4}$ are almost identical.

3.4.2 The effect of curvature

As mentioned in the previous section the curved walls in a Cartesian grid will take into account with the penalization method. For validation of the method the same benchmark problem, i.e., dipole collision with straight wall, will be considered in a penalized cavity to compare the results with classical boundary conditions (Dirichlet and/or Neumann) as explained in Section 2.6. All of the computations in this section are performed with the uniform multigrid solver because of its efficiency in terms of CPU-time. In the first step, for $Re = 1000$, the simulations will be performed for quantitative comparisons of total energy $E(t)$, total enstrophy $Z(t)$ and total palinstrophy $P(t)$. The comparisons will be in a same geometry, with the same initial condition. Two category for simulation of a fixed wall will be considered; first the classical no-slip an no-penetration conditions in boundaries of Cartesian grid, second, the velocity in a marginal area of a slightly larger cavity will imposed to zero with penalization method. A schematic representation of the penalized cavity, i.e., the geometry and initial condition are illustrated in Fig. 3.19 (a) and (b), respectively. Comparisons of total energy, total enstrophy and total palinstrophy between the two



(a) Total enstrophy (b) Total palinstrophy*

Figure 3.16: Convergence study for the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ toward the data from Clercx with the present finite difference computation with uniform grid solver for Reynolds 1000 and maximum grid level $J = 8, 9, 10, 11$ in each direction.

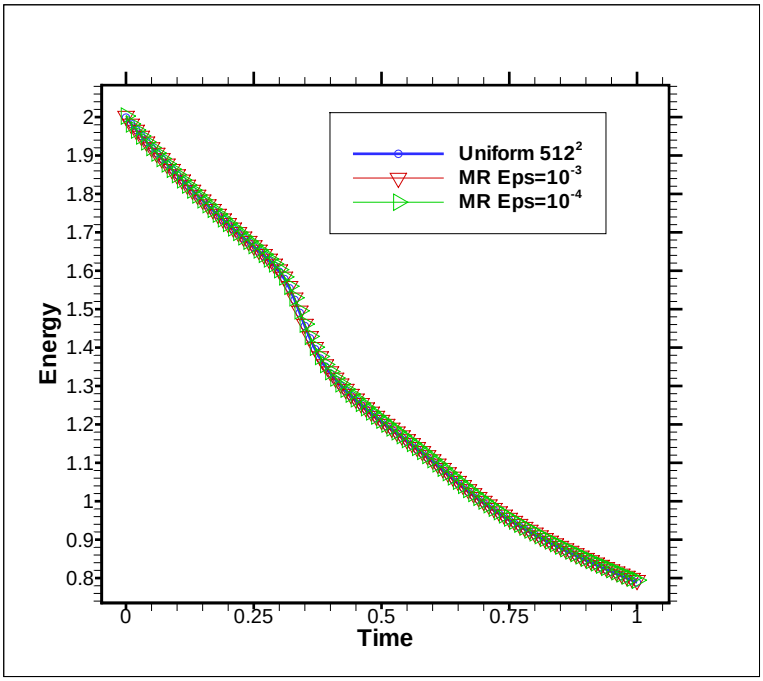
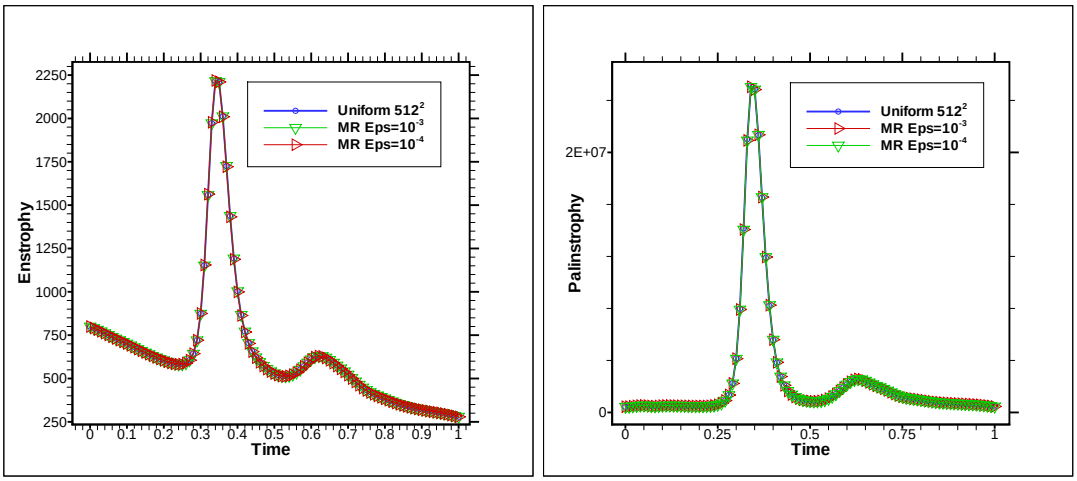


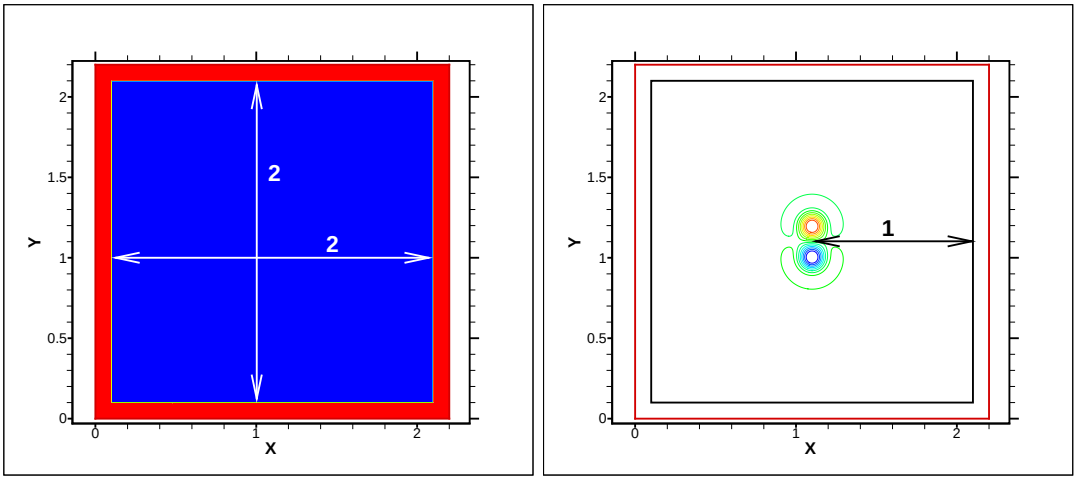
Figure 3.17: Comparison of the total energy $E(t)$ between the uniform grid solver and the multiresolution computation with thresholds, $\epsilon = 10^{-3}$ and $\epsilon = 10^{-4}$ for Reynolds 1000 and maximum grid level $J = 9$ in each direction for all simulations.



(a) Total enstrophy (b) Total palinstrophy*

Figure 3.18: Comparisons of the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ between the uniform grid solver and the multiresolution computation with thresholds, $\epsilon = 10^{-3}$ and $\epsilon = 10^{-4}$ for Reynolds 1000 and maximum grid level $J = 9$ in each direction for all simulations.

cited methods for simulation of a rigid wall for $Re = 1000$ with a maximum grid level of $J = 10$ in each direction are illustrated in Fig. 3.20, Fig. 3.21 (a) and (b), respectively. The small deviation in the total energy, which is almost invisible, is negligible and below 3%. For both total enstrophy and total palinstrophy a considerable deviation can be observed, which can be explained by the first order nature of implementing a solid wall via the volume penalization method. So one must use a very fine grid for obtaining more accurate results with this method.
After the procedure of validation, for considering the effect of curvature we will continue with Reynolds 10000 because the flow field is more sensitive to the curvature of the hitting wall. Thus this Reynolds number will be used for studying the effect of curvature in the collision of a dipole with concave and convex walls, even if a grid with 1025^2 points is not sufficient to fully resolved all the spatial scales who are present in the flow. But the numerical dissipation of the second-order central method will compensate the effect of eliminated scales to have a stable simulation. For having a reference solution to compare the results of the penalization method for curved walls, first the dipole collision with a straight penalized wall at Reynolds 10000 will be simulated. A schematic representation of the penalized cavity, i.e., the geometry and the initial condition are illustrated in Fig. 3.22 (a) and (b), respectively. To obtain a qualitative idea about the flow and its future destiny the evolution and collision of dipole with a straight penalized wall from $t = 0$ to $t = 2$ for Reynolds 10000 is illustrated in Fig. 3.23 and Fig. 3.24. The simulations are performed with a multigrid solver over a uniform grid with maximum level $J = 10$ in each direction. To study the effect of curvature two configurations will be considered. First the collision of a dipole with a convex wall with radius $R = 1.5$ where the center of the convex wall is placed at $x_c = 3.1$, $y_c = 1.0$. The initial vortices are placed at distance equal to one from the right wall in all of the simulations. Second the collision with a concave



(a) Mask function (b) Initial condition

Figure 3.19: Schematic representation of penalized cavity (blue represents the fluid domain) where in solid domain (represented by red) $\eta = 10^{-3}$ is imposed. The geometry and the initial vorticity distribution are the same as the benchmark.

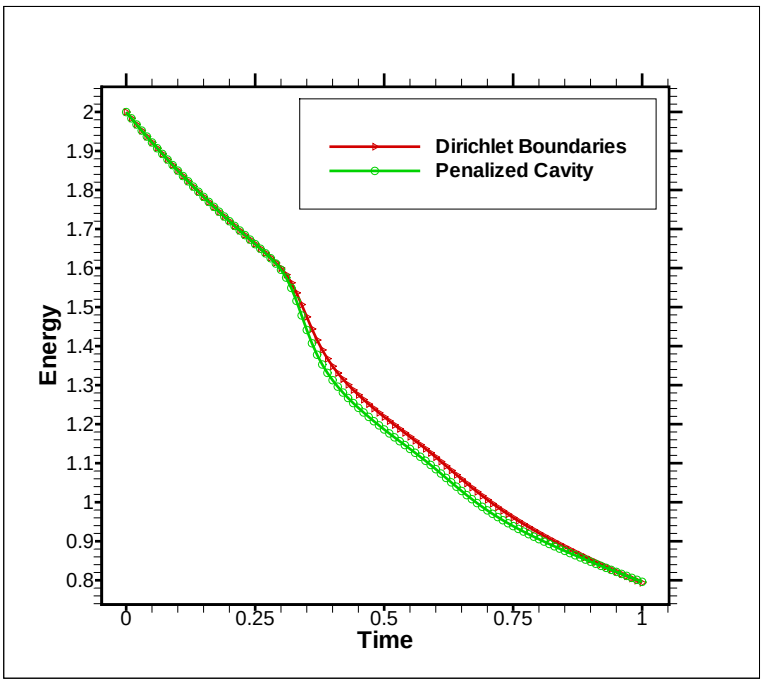
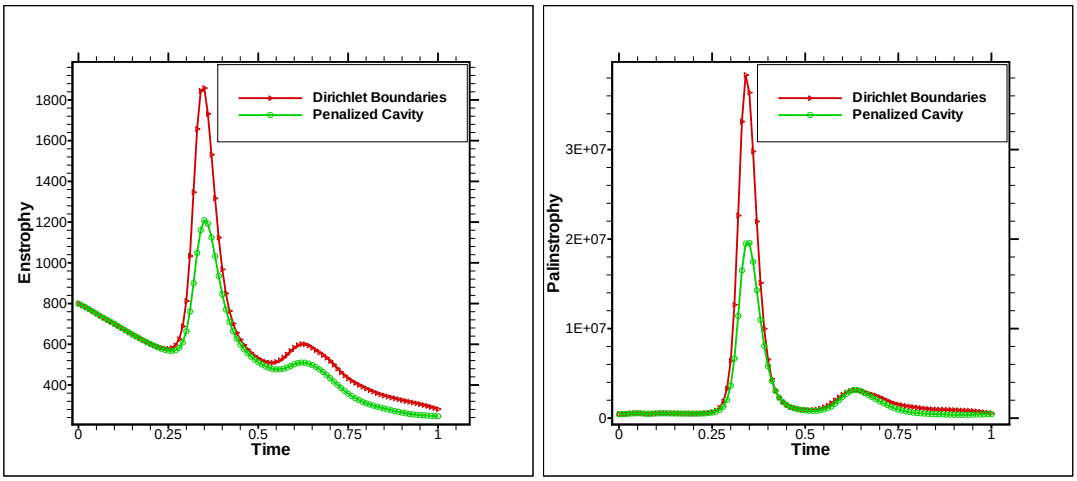


Figure 3.20: Comparison of total energy $E(t)$ of a dipole collision with a no-slip and no-penetration straight wall by imposing classical (Dirichlet and/or Neumann) boundary conditions and a penalized wall, for Reynolds 1000 and maximum grid level $J = 10$ in each direction.

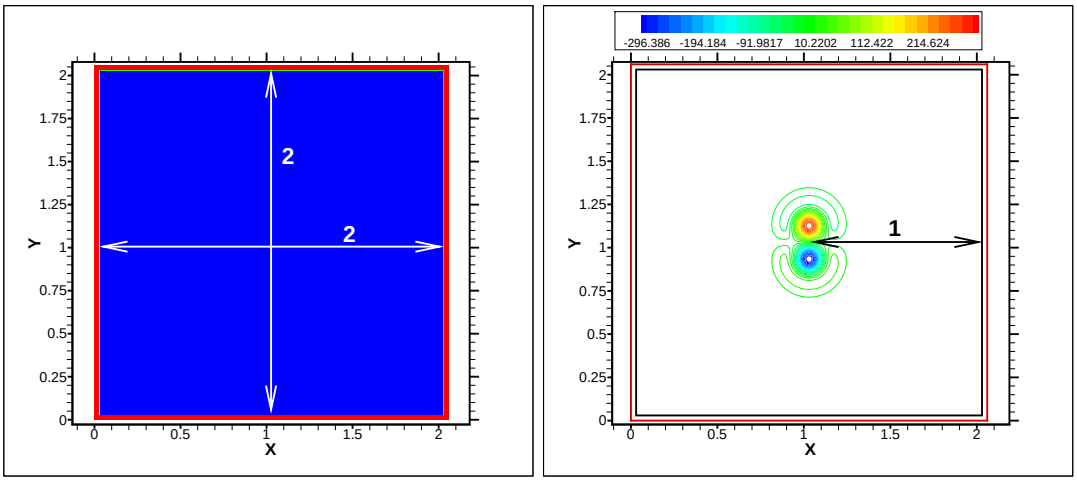


(a) Total enstrophy (b) Total palinstrophy

Figure 3.21: Comparison of the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ of a dipole collision with a no-slip and no-penetration straight wall by imposing classical (Dirichlet and Neumann) boundary conditions and a penalized wall, for Reynolds 1000 and maximum grid level $J = 10$ in each direction.

wall with radius $R = 1.5$ will be examined where its center is placed at $x_c = 0.5$, $y_c = 1.0$. The corresponding mask functions and initial conditions are illustrated in Fig. 3.25 and Fig. 3.26, respectively. For having a qualitative idea about the collision of the dipole with a convex wall, the flow evolution from the initial condition $t = 0$ up to $t = 2$, is illustrated in Fig. 3.27 and Fig. 3.28 for Reynolds 10000. This will be useful for a qualitative comparison with dipole collision with straight wall. The computations were done with the same solver over the same grid points, i.e., 1025^2. Next the evolution and the collision of the dipole with a concave wall from the initial condition $t = 0$ to $t = 2$, is illustrated in Fig. 3.29 and Fig. 3.30 for Reynolds 10000. Comparisons of total energy, enstrophy and palinstrophy between collision with a straight wall, a convex and a concave wall, with $R = 1.5$, for Reynolds 10000 and maximum grid level $J = 10$ in each direction, are shown in Fig. 3.31 and Fig. 3.32 (a) and (b), respectively.

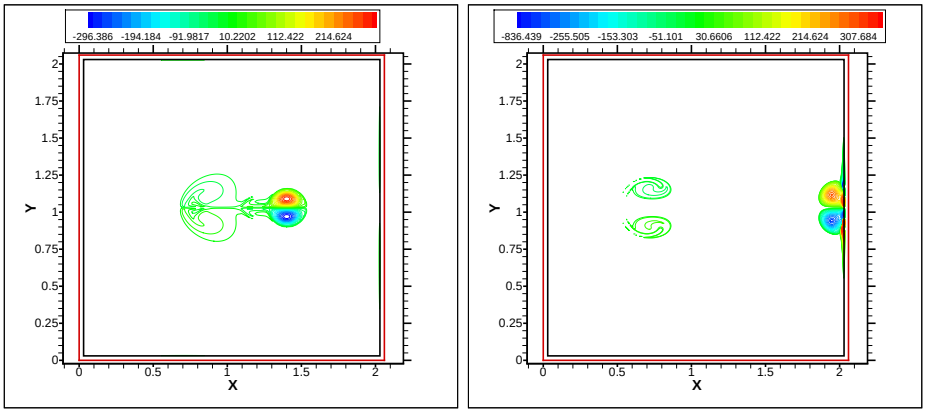
In the total energy evolution, the curve corresponding to the straight wall lies between those of the convex and concave walls up to $t = 1.1$, then it falls below the those of curved walls. But the total energy evolution for the concave wall lies always above the others, which can be explained as follows; For total enstrophy and palinstrophy, the curve for the concave wall usually lies below the others, which can be justified by considering the fact that in that case the surface is minimum, so enstrophy production at walls is reduced with respect to the other cases. Thus by considering Eq. (3.16) the results are qualitatively satisfactory. Some heuristic general features of the flow for different configurations can be summarized as follows; The time of the first collision with the right wall at sufficient high Reynolds numbers, independent of the curvature of the hitting wall is more or less the same, approximately equal to $t_{col} = 0.3$. There will be a rise in the vorticity of small newly generated vortices after



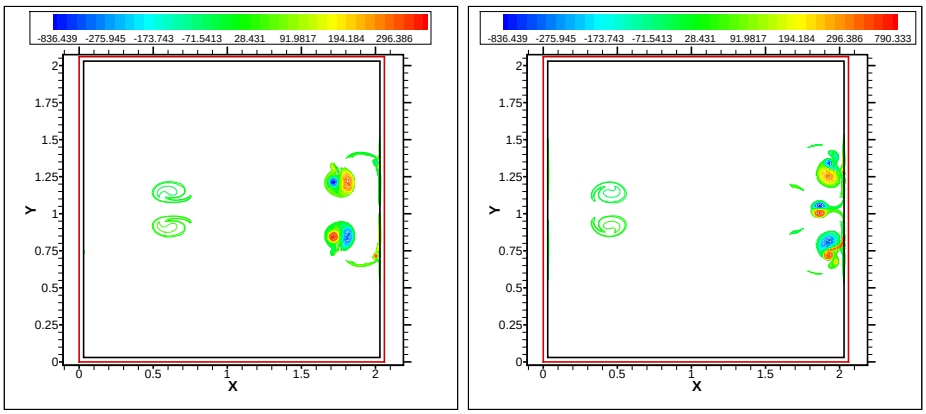
(a) Mask function (b) Initial condition

Figure 3.22: Schematic representation of the penalized cavity (blue represents the fluid domain) where in solid domain (represented by red) $\eta = 10^{-3}$ is imposed. The geometry and initial condition (initial vorticity distribution) are the same as the benchmark.

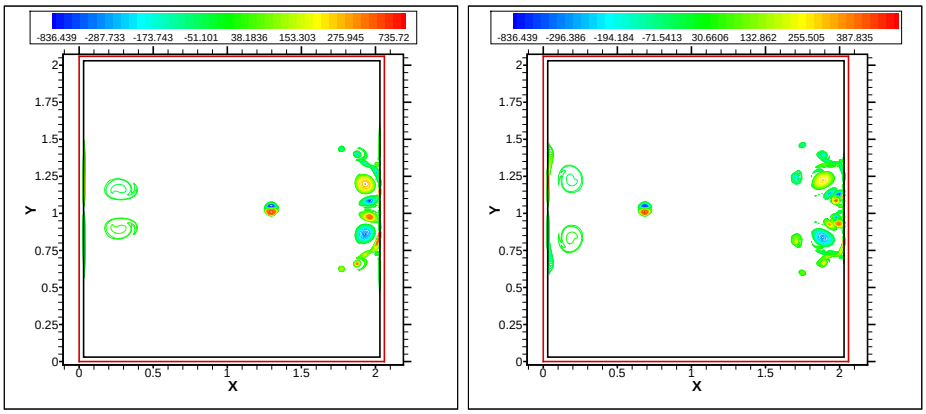
the collision with the walls, this is also independent of the curvature of the hitting wall. The destiny of the small newly generated dipole reflecting from the right wall after the second collision of the dipoles with right wall will be different when arriving to the left wall. One can see the division of this dipole into two vortices in the concave and convex test cases, while it will remain attached in the case of straight wall because it is stronger. At $t = 2$ in the case of collision with a straight wall the number of vortices are about $n = 13$, while in the cases of concave or convex walls they are about $n = 8$. One can see different phenomena in the field such as attachment to and detachment of vortices from the walls, merging of counter-rotating vortices, creation and dissipation of vortex sheets (in two dimensions they will be filaments), creation of new small vortices near the wall with high rotational speed and dissipation of the vortices during the time. Also collision of vortices and dipoles of different sizes with the walls and themselves, interaction of vortices, new dipoles or new isolated vortices moving with different self induced velocities, the effort of vortices to become circular and symmetric (isotropy) especially in the regions far from the wall to minimize the potential. The creation of anisotropy near solid walls can also be seen. Another point to be noted is that the vortices are very persistent and their life time is much longer in comparison to three-dimensional flows, in which the vortex stretching mechanism is responsible for the fast decay of vortices by decreasing their size and increasing the vorticity in the center line of the vortex tubes thus leading to the direct energy cascade. In two-dimensional flows, the dominant mechanism to destroy a vortex is the dissipation which becomes small for high Reynolds numbers. The simulations also exhibit a symmetry of the flow with respect to the $y = 1$ axis, up to t=4. Then the symmetry is broken due to the instability of the flow.



(a) $t = 0.15$ (b) $t = 0.3$

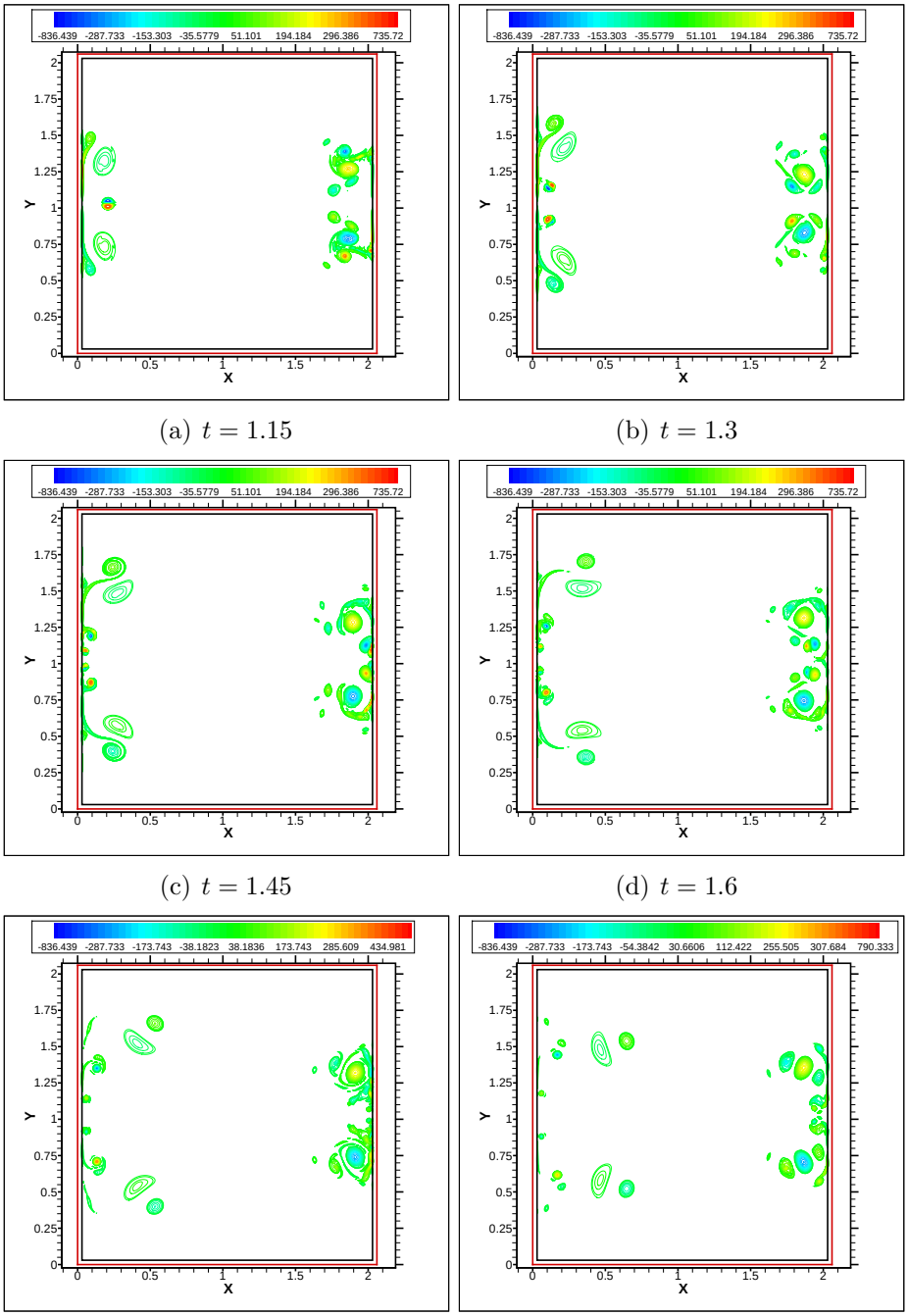


(c) $t = 0.45$ (d) $t = 0.6$



(e) $t = 0.8$ (f) $t = 1.0$

Figure 3.23: The evolution and collision of the dipole (vortices represented by colored isolines) with straight penalized walls for Reynolds 10000. Maximum grid level $J = 10$ in each direction and $\eta = 10^{-3}$ in the penalized margin is imposed. The black lines represent the boundaries of the penalized domain.

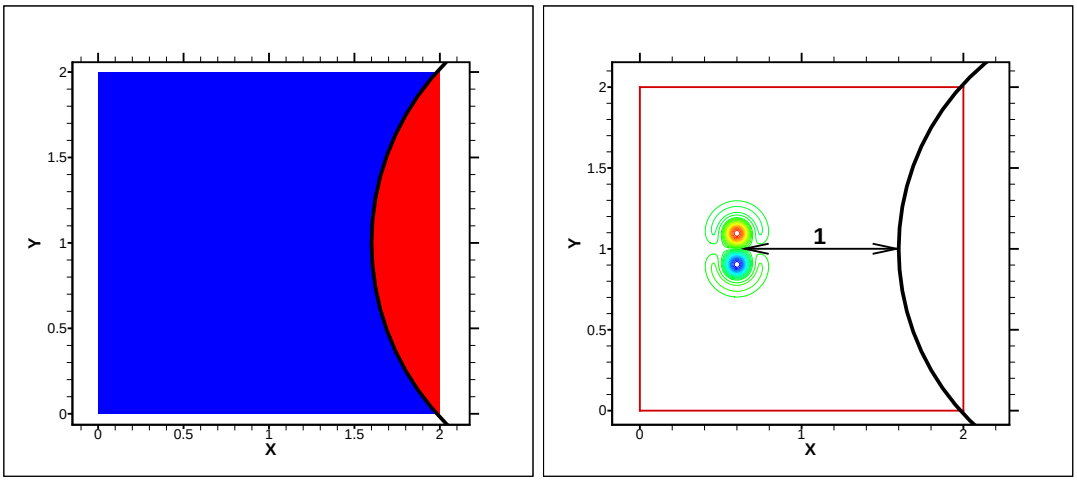


(a) $t = 1.15$ (b) $t = 1.3$

(c) $t = 1.45$ (d) $t = 1.6$

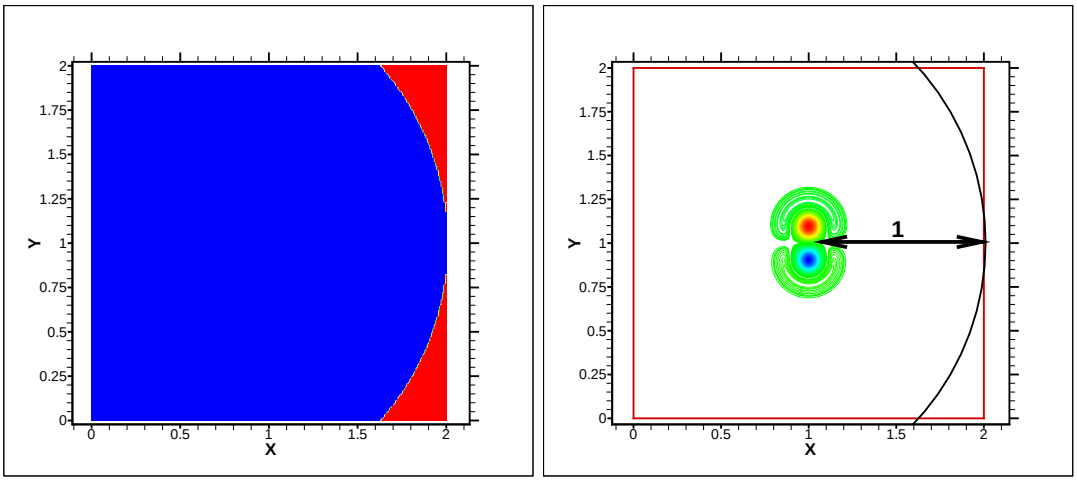
(e) $t = 1.8$ (f) $t = 2.0$

Figure 3.24: The evolution and collision of the dipole (vortices represented by colored isolines) with straight penalized walls for Reynolds 10000. Maximum grid level $J = 10$ in each direction and $\eta = 10^{-3}$ in the penalized margin is imposed. The black lines represent the boundaries of the penalized domain.



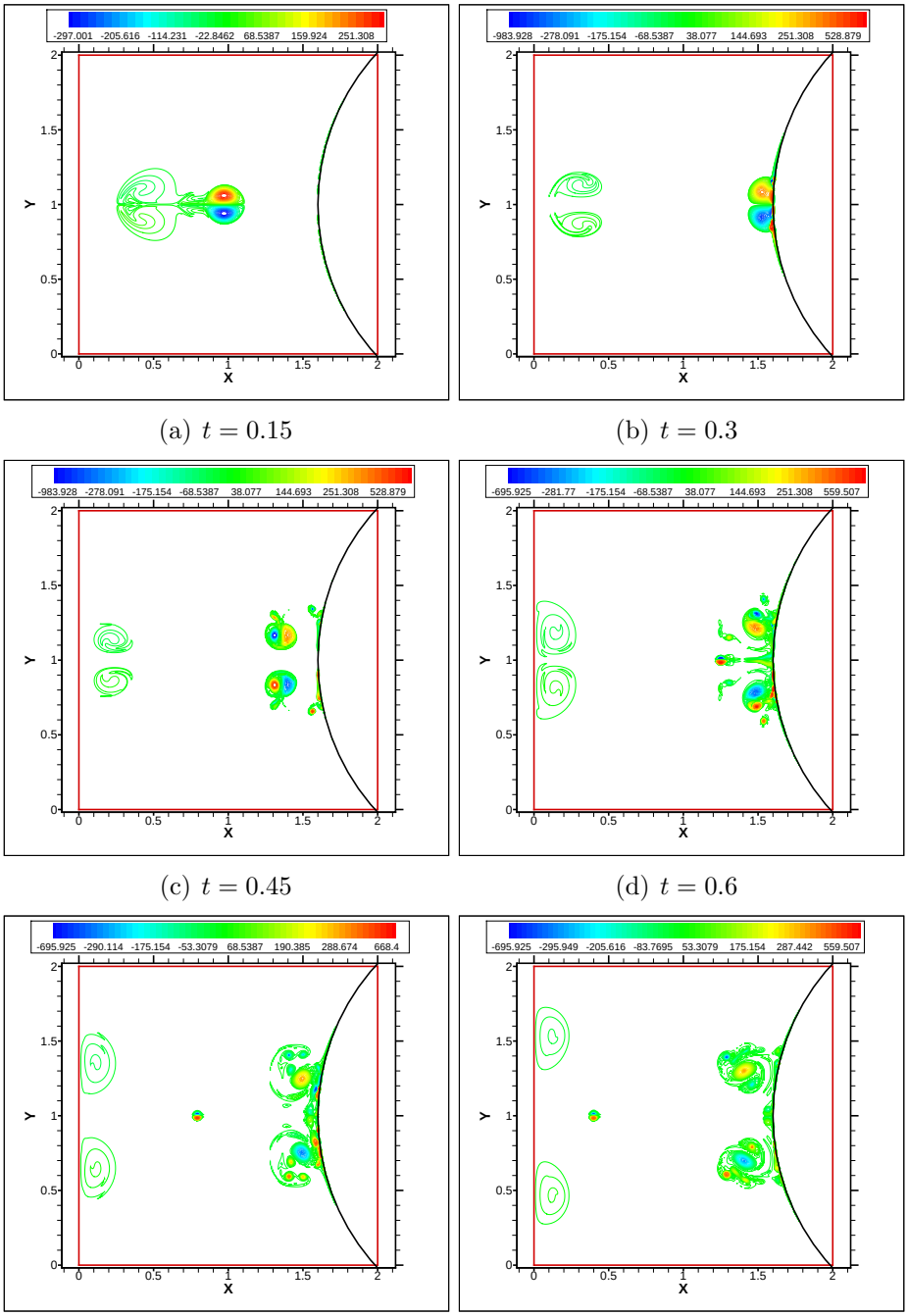
(a) Mask function (b) Initial condition

Figure 3.25: Mask function (blue represents the fluid domain) with $\eta = 10^{-3}$ inside the convex wall, with $R = 1.5$, $x_c = 3.1$, $y_c = 1.0$, the initial vortices are placed at distance equal to one from the right wall.



(a) Mask function (b) Initial condition

Figure 3.26: Mask function (blue represents the fluid domain) with $\eta = 10^{-3}$ inside the concave wall, with $R = 1.5$, $x_c = 0.5$, $y_c = 1.0$, the initial vortices are placed at distance equal to one from the right wall.

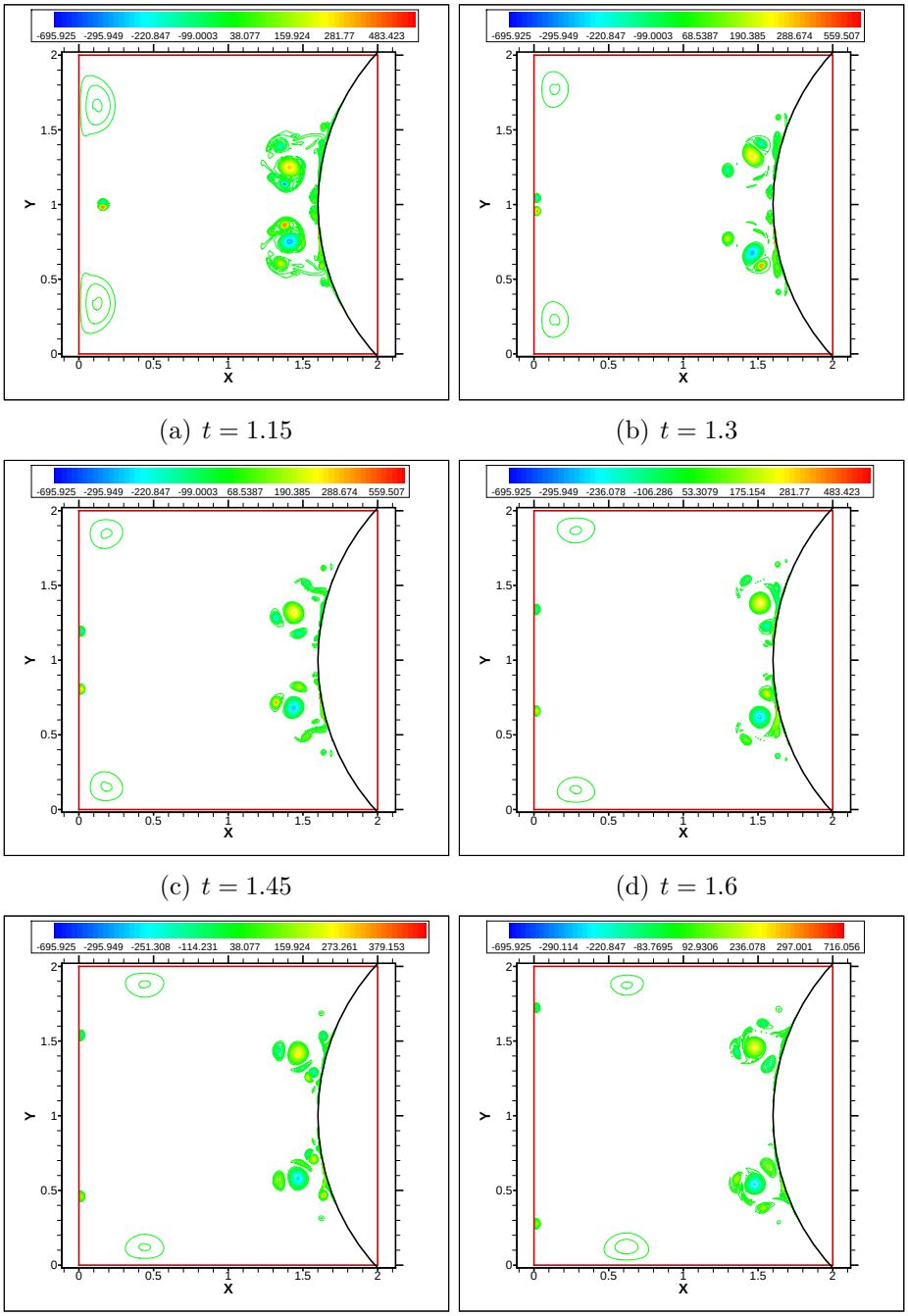


(a) $t = 0.15$ (b) $t = 0.3$

(c) $t = 0.45$ (d) $t = 0.6$

(e) $t = 0.8$ (f) $t = 1.0$

Figure 3.27: The evolution and collision of the dipole (the vortices represented by colored isolines) with a convex wall for Reynolds 10000. The center of the mask is placed at ($x_c = 3.1$, $y_c = 1.0$) with $R = 1.5$, maximum grid level $J = 10$ in each direction is imposed.

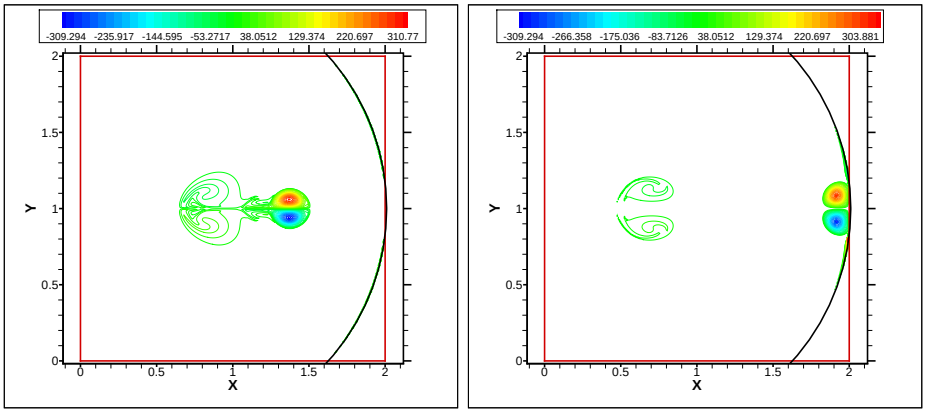


(a) $t = 1.15$ (b) $t = 1.3$

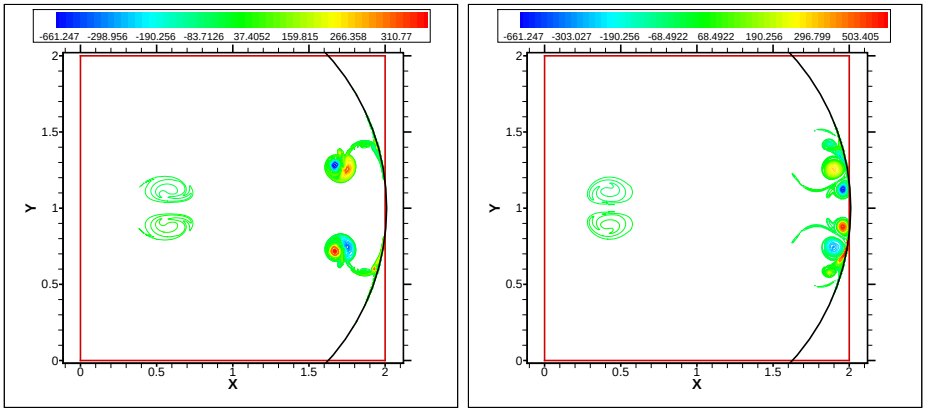
(c) $t = 1.45$ (d) $t = 1.6$

(e) $t = 1.8$ (f) $t = 2.0$

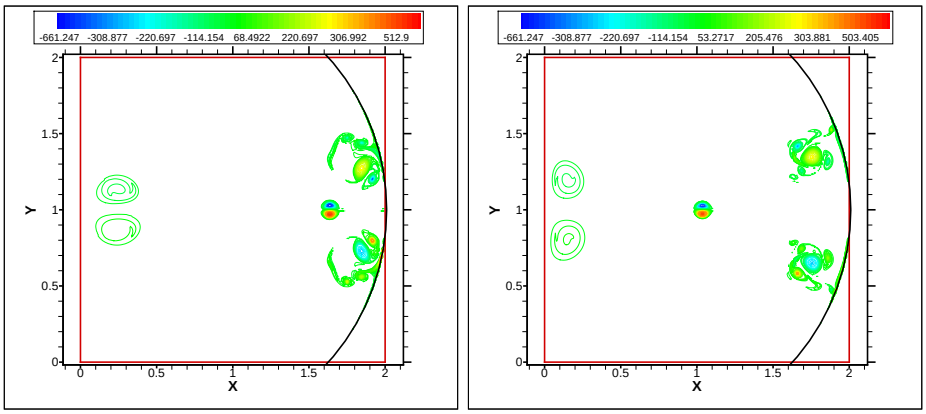
Figure 3.28: The evolution and collision of the dipole (the vortices represented by colored isolines) with a convex wall for Reynolds 10000. The center of the mask is placed at ($x_c = 3.1$, $y_c = 1.0$) with $R = 1.5$, maximum grid level $J = 10$ in each direction is imposed.



(a) $t = 0.15$ (b) $t = 0.3$

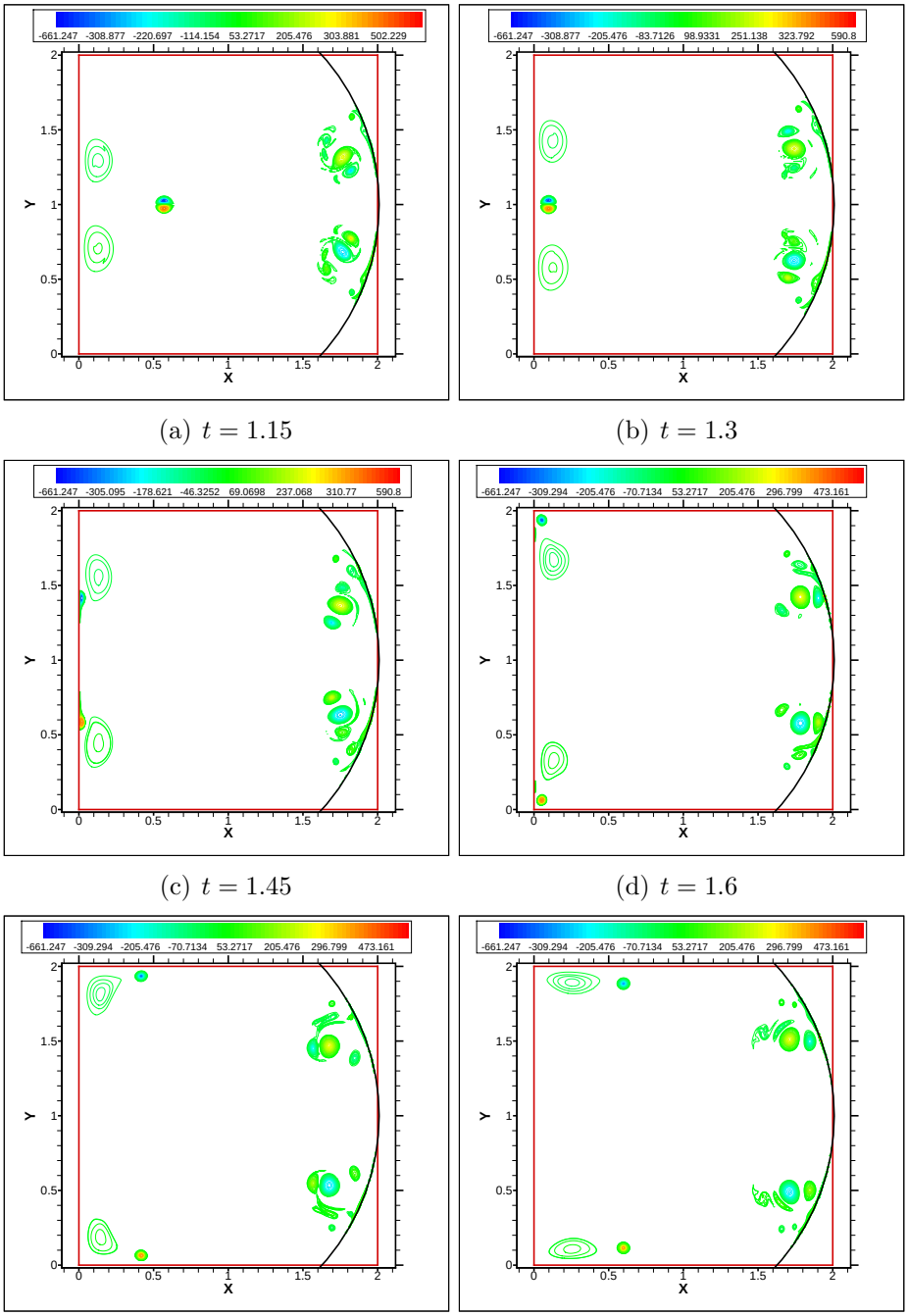


(c) $t = 0.45$ (d) $t = 0.6$



(e) $t = 0.8$ (f) $t = 1.0$

Figure 3.29: The evolution and collision of the dipole (the vortices represented by colored isolines) with a concave wall for Reynolds 10000. The center of the mask is placed at ($x_c = 0.5$, $y_c = 1.0$) with $R = 1.5$, maximum grid level $J = 10$ in each direction is imposed.



(a) $t = 1.15$ (b) $t = 1.3$

(c) $t = 1.45$ (d) $t = 1.6$

(e) $t = 1.8$ (f) $t = 2.0$

Figure 3.30: The evolution and collision of the dipole (the vortices represented by colored isolines) with a concave wall for Reynolds 10000. The center of the mask is placed at ($x_c = 0.5$, $y_c = 1.0$) with $R = 1.5$, maximum grid level $J = 10$ in each direction is imposed.

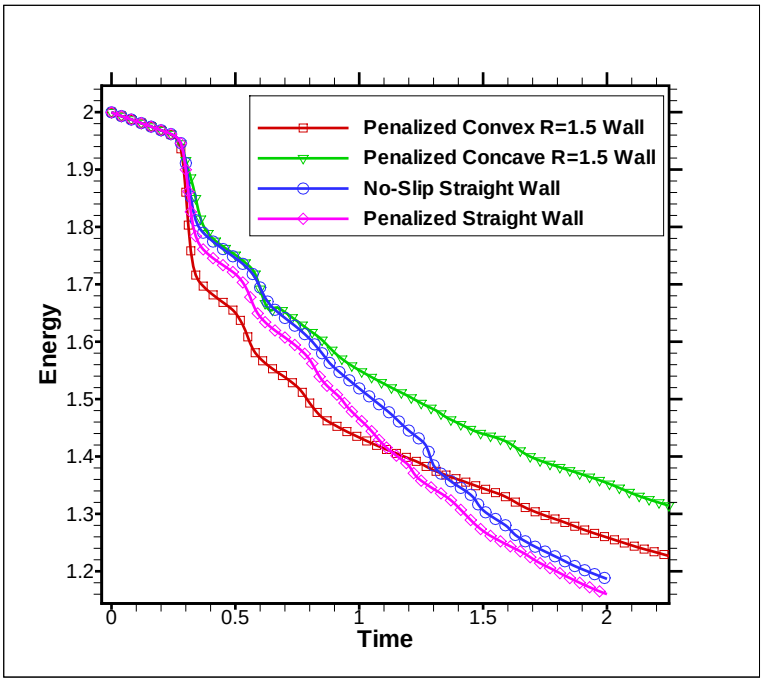
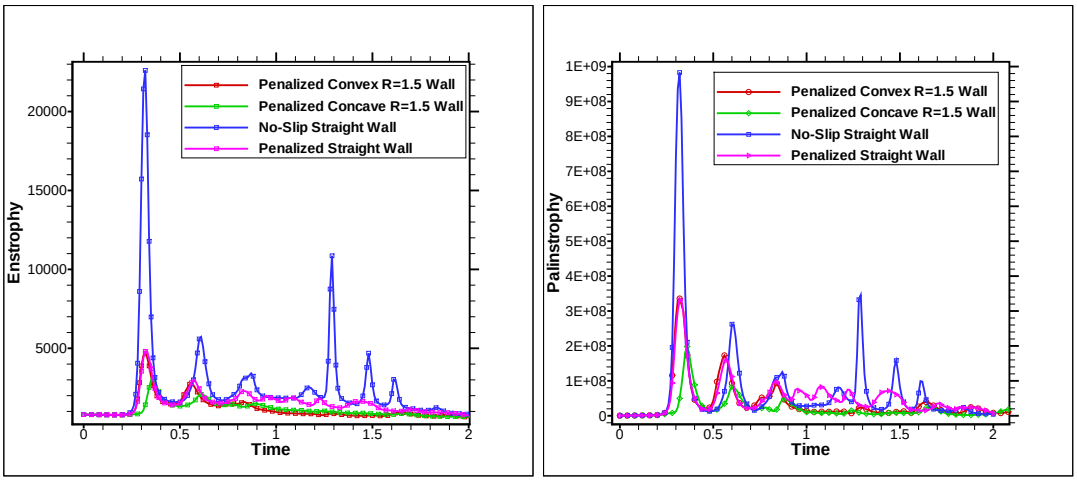


Figure 3.31: Comparison of the total energy $E(t)$ of a dipole collision with a straight wall by imposing classical no-slip and no-penetration (Dirichlet and Neumann) boundary conditions and a penalized straight, convex and concave walls, $R = 1.5$, for Reynolds 10000 and maximum grid level $J = 10$ in each direction.



(a) Total enstrophy (b) Total palinstrophy*

Figure 3.32: Comparison of the total enstrophy $Z(t)$ and the total palinstrophy $P(t)$ of a dipole collision with a straight wall by imposing classical no-slip and no-penetration (Dirichlet and Neumann) boundary conditions and a penalized straight, convex and concave walls, $R = 1.5$, for Reynolds 10000 and maximum grid level $J = 10$ in each direction.

Chapter 4

Conclusions and perspectives

In the current master thesis a self-adaptive code for direct numerical simulation of two-dimensional unsteady incompressible flows has been developed. The new algorithm is based on a central second-order finite difference discretization of the governing equations coupled with Harten's point value multiresolution analysis. The multiresolution analysis uses cubic interpolating wavelet transform for grid adaptation. This kind of wavelet transform will impose the necessity of using Cartesian uniform grids. The use of these types of grids will lead to good accuracy, efficiency and simplicity of the solver. Alongside with these advantages, the implementation of curved boundaries in Cartesian grids is a dilemma. The remedy for this problem is to use the idea of immersed boundary methods, e.g., volume penalization method. The multiresolution analysis allows to reduce the number of active grid points significantly, nevertheless the analytical accuracy order of the underlying numerical scheme is preserved. For time integration explicit Runge-Kutta methods of different order are implemented. First the algorithm was applied to the one-dimensional Burgers equation and the errors in time and in space have been assessed in detail. Then the method was extended to the incompressible two-dimensional Navier-Stokes equations expressed in vorticity-stream function formulation. Therewith the near wall behavior of two-dimensional vortical flows has been studied for different geometries, i.e., a dipole-wall collision with either straight or curved walls. For implementation of curved boundaries in a Cartesian grid the volume penalization method has been used. This method treats solid obstacles or walls as porous media, no-slip and no-penetration boundary conditions can thus be enforced by using permeability coefficient of the solid region which tend to zero. The obtained results show that the CPU-time of the adaptive simulations can be significantly reduced with respect to simulations on a regular uniform grid. The only parameter to be adjusted in the adaptive solver is the threshold parameter ϵ, which directly controls the compression rate, hence the introduced error in the nonlinear filtering procedure. The numerical experiments showed that for the considered one and two dimensional problems with maximum grid level less than $J = 12$ the proper threshold parameter is $\epsilon = 10^{-3}$. The average number of active points over the base grid is less than 10% for one-dimensional problems, thus the compression rate of grid points is usually more than 90%, which is promising. The time of the multiresolution computations for one-dimensional problems is reduced by more than a factor two

with respect to the computations on a uniform grid, taking into account the overhead necessary for the multiresolution algorithm. For higher resolutions further reduction in CPU-time are expected. The average number of active points over the base grid for two-dimensional problems is higher than that of one-dimensional problems due to safety zone, which is necessary to account for the translation of the solution and the generation of finer scales. In two dimensions 16 points for each active point must be added as safety zone, four at the same and four at one finer scale, while in one space dimension only 4 points are necessary as safety zone, two at the same and two at one finer scale. For the two-dimensional computations the compression rate is typically about 25%, which is thus not so high. However, the CPU time of the multiresolution computations is at least one sixth in comparison with the uniform solver (for maximum grid level $J = 10$ in each direction). This is due to presence of the elliptic solver, thus an extension to this work will be development of a multigrid solver or Krylov subspace methods for the multiresolution algorithm, to accelerate further the elliptic part of the method. Another suggestion is to develop a wavelet transform based on fifth-order interpolation which will lead to higher compression rates in comparison to the cubic interpolation. This is the case especially for incompressible flows where the flow fields are much smoother than that of compressible flows where shocks may occur in the field. The volume penalization method has been introduced into the finite difference solvers, both on the uniform grid solver and the adaptive one, to be able to compute flows in complex geometries, like curved walls. Comparing the solutions obtained via the penalization method with that of classical no-slip and no-penetration boundary conditions showed that for the dipole-wall collision with a straight wall, for Reynolds 1000 with 1024^2 grid point, we found an average deviation of less than 2% and a maximum deviation of 6% in the total energy evolution. For evolution of the total enstrophy and the total palinstrophy the difference between the two methods of applying the boundary conditions is however larger, a maximum difference (coincide with the first collision) of 30% in total enstrophy and 50% in total palinstrophy evolutions is seen. The explanation is given by the first-order accuracy of the volume penalization method at the wall. Hence small permeability parameters are necessary and fine grids near boundaries have to be used. Furthermore it was shown that the finite difference computations with classical boundary conditions converge towards the pseudo-spectral computations of Clercx et al. For Reynolds 1000 it was found that the grid resolution of $J = 11$ in each direction yields to reasonable results for energy, however some difference, 12% in enstrophy and in particular 20% for palinstrophy are still present. It is anticipated that for grid resolutions of $J = 12$ in each direction these differences will become much smaller, but the computational time with out parallel processing is currently too prohibitive. Extension of the volume penalization method to a high order (e.g., second order) immersed boundary method (IBM) will be another forward step. One interesting perspective is the identification and the tracking of the vortex cores in the flow field to study more quantitatively the details of the vortex dynamics. Extending the program to include a dynamic and distributed memory option for parallel computations with message passing interface (MPI) will be another step of the future investigation by using either hash tables or tree data structures. For helpful discussions in this area see [51] and [53].

Bibliography

[1] A. Thom, The flow past circular cylinders at low speeds. Proceedings of the Royal Society, London, Section A, 141, 651-669, 1933.

[2] A. J. Chorin, A numerical method for solving incompressible viscous flow problems. Journal of Computational Physics, Vol. 2, 12-26, 1967.

[3] A. J. Chorin, Numerical solution of Navier-Stokes equations. Mathematics of Computation, Vol. 22 (104), 745-62, 1968.

[4] W. R. Briley, A numerical study of laminar separation bubbles using the Navier-Stokes equations. Journal of Fluid Mechanics, Vol. 47, 713-736, 1971.

[5] J. D. A. Walker, The boundary layer due to a rectilinear vortex. Proceedings of the Royal Society, Series A, 359, 167, 1978.

[6] R. H. Kraichnan and D. Montgomery, Two-dimensional turbulence. Reports on Progress in Physics, Vol. 43, 1980.

[7] E. Arquis et J.P. Caltagirone, Sur les conditions hydrodynamiques au voisinage d'une interface milieu fluid - milieux poreux: application à la convection naturelle. Comptes Rendus de l'Academie des Sciences, Paris, 2(299), 1-4, 1984.

[8] P. Goupillaud, A. Grossmann and J. Morlet, Cycle-octave and related transforms in seismic signal analysis. Journal of Applied Geophysics, Vol. 23, Issue 1, 85-102, 1984.

[9] J. Kim and P. Moin, Application of a fractional-step method to incompressible Navier-Stokes equations. Journal of Computational Physics, Vol. 59, 308-323, 1985.

[10] K. Z. Meth, A vortex and finite difference hybrid method to compute the flow of an incompressible, inviscid fluid past a semi-infinite plate. PhD thesis, NYU, 1988.

[11] J. Liandrat and P. Tchamitchian, Resolution of the 1D regularized burgers equation using a spatial wavelet approximation. Technical Report NASA Contractor Report 187480, NASA Langley Research Center, Hampton VA, 1990.

[12] P. Orlandi, Vortex dipole rebound from a wall. Physics of Fluids, A 2, 1429, 1990.

[13] S. E. Rogers, D. Kwak and C. Kiris, Steady and unsteady solutions of the incompressible Navier-Stokes equations. AIAA Journal, Vol. 29(4), 603-610, 1991.

[14] E. A. Coutsias and J. P. Lynov, Fundamental interactions of vortical structures with boundary layers in two-dimensional flows. Physica D, Vol. 51, 482, 1991.

[15] V. J. Peridier, F. T. Smith and J. D. A. Walker, Vortex-induced boundary layer separation. Part 1. The unsteady limit problem $Re \rightarrow \infty$. Journal of Fluid Mechanics, Vol. 232, 99, 1991.

[16] V. J. Peridier, F. T. Smith and J. D. A. Walker, Vortex-induced boundary layer separation. Part 2. Unsteady interacting boundary-layer theory. Journal of Fluid Mechanics, Vol. 232, 133, 1991.

[17] D. L. Donoho, Interpolating wavelet transforms. Technical Report 408, Department of Statistics, Stanford University, 1992.

[18] H. Press, A. Teukolsky, T. Vetterling and P. Flannery, Numerical Recipes in Fortran 77. (The Art of Scientific Computing) 2nd Edition, Cambridge University Press, ISBN: 0-521-43064-X, 1992.

[19] T. Y. Hou and B. T. R. Wetton, Convergence of a finite difference scheme for the Navier-Stokes equations using vorticity boundary conditions. SIAM Journal on Numerical Analysis, Vol. 29, 615-639, 1992.

[20] A. Harten, Multiresolution Representation of data: A general framework. SIAM Journal on Numerical Analysis, Vol. 33, No. 3, 1205-1256, 1996.

[21] B. L. Bihari and A. Harten, Multiresolution schemes for the numerical solution of 2-D conservation laws. SIAM Journal on Scientific Computing, Vol. 18, No. 2, 315-354, 1997.

[22] W. Sweldens, The lifting scheme: A construction of second generation wavelets. SIAM Journal on Mathematical Analysis, Vol. 29, No. 2, 511-546, 1998.

[23] P. Angot, C.-H. Bruneau and P. Fabrie, A penalization method to take into account obstacles in viscous flows. Numerische Mathematik, Vol. 81, 497-520, 1999.

[24] M. Holmstrom, Solving hyperbolic PDEs using interpolation wavelets. SIAM Journal on Scientific Computing, Vol. 21, No. 2, 405-420, 1999.

[25] J. M. Powers and S. Paolucci, Manifold methods for energetic materials. Presented at the IMA workshop on High-Speed Combustion in Gaseous and Condensed-Phase Energetic Materials, University of Minnesota, 11 November 1999.

[26] J. Bodeau, G. Riboulet and T. Roncalli, Non uniform grids for PDE in finance. Rapport de Groupe de Rechereche Opérationnelle, France, 2000.

[27] O. V. Vasilyev and C. Bowman, Second generation wavelet collocation method for the solution of partial differential equations. Journal of Computational Physics, vol. 165, 660-693, 2000.

[28] H. Lomax, T. H. Pulliam and D. W. Zingg, Fundamentals of Computational Fluid Dynamics. Springer-Verlag, Berlin, ISBN: 3-540-41607-2, 2001.

[29] H. J. H. Clercx and G. J. F. van Heijst, Dissipation of kinetic energy in two-dimensional bounded flows. Physical Review E, Vol. 65, 066305, 2002.

[30] A. V. Obabko and K. W. Cassel, Navier-Stokes solutions of unsteady separation induced by a vortex. Journal of Fluid Mechanics, Vol. 465, 99, 2002.

[31] C. Wang and J. G. Liu, Analysis of finite difference schemes for unsteady Navier-Stokes equations in vorticity formulation. Springer, Numerische Mathematik, Vol. 91, 543-576, 2002.

[32] P. A. Durbin and G. Iaccarino, An approach to local refinement of structure grids. Journal of Computational Physics, Vol. 181, Issue 2, 639-653, 2002.

[33] O. Roussel, Développement d'un algorithme multirésolution adaptatif tridimensionnel pour les équations aux dérivées partielles paraboliques. Application l'étude des instabilités thermo-diffusives de flamme. PhD thesis, Université d'Aix-Marseille, 2003.

[34] O. V. Vasilyev, Solving multi-dimensional evolution problems with localized structures using second generation wavelets. International Journal in Computational Fluid Dynamics, special issue on High-resolution Methods in Computational Fluid Dynamics, Vol. 17, No. 2, 151-168, 2003.

[35] A. Naguib and M. Koochesfahani, On wall-pressure sources associated with the unsteady separation in a vortex-ring/wall interaction. Physics of Fluids, Vol. 16(7), 2613-2622, 2004.

[36] D. Kwak, C. Kiris and C. S. Kim, Computational challenges of viscous incompressible flows. Computers and Fluids, Vol. 34, 283-299, 2005.

[37] O. V. Vasilyev and N. K.-R. Kevlahan, An adaptive multilevel wavelet collocation method for elliptic problems. Journal of Computational Physics, Vol. 206, 412-431, 2005.

[38] D. Rempfer, On boundary conditions for incompressible Navier-Stokes problems. Applied Mechanics Reviews, Vol. 59, Issue 3, 107-125, 2006.

[39] H. Ding and C. Shu, A stencil adaptive algorithm for finite difference solution of incompressible viscous flows. Journnal of Computational Physics, Vol. 214, Issue 1, 397-420, 2006.

[40] J. L. Guermond, P. Minev and J. Shen, An overview of projection methods for incompressible flows. Computer Methods in Applied Mechanics and Engineering, Vol. 195, 6011-6045, 2006.

[41] K. Schneider and M. Farge, Wavelets: Mathematical Theory. Encyclopedia of Mathematical Physics, Elsevier, 426-438, 2006.

[42] H. J. H. Clercx and C.-H. Bruneau, The normal and oblique collision of a dipole with a no-slip boundary. Computers and Fluids, Vol. 35, 245, 2006.

[43] J. M. Alam, N. K.-R. Kevlahan and O. V. Vasilyev, Simultaneous space-time adaptive wavelet solution of nonlinear parabolic differential equations. Journal of Computational Physics, Vol. 214, 829-857, 2006.

[44] D. Kolomenskiy, Simulation numérique direct des écoulement tourbillonaires autour des obstacles mobiles. Stage de Master Recherche Mécanique, Physique et Modélisation, Université d'Aix-Marseille, 2007.

[45] W. Kramer, H. J. H. Clercx and G. J. F. van Heijst, Vorticity dynamics of a dipole colliding with a no-slip wall. Physics of Fluids, Vol. 19, 126603, 2007.

[46] A. A. Siddiqui and M. T. Mustafa, Wavelet optimized finite difference method with non-static re-gridding. Applied Mathematics and Computation, Vol. 186, 203-211, 2007.

[47] G. H. Keetels, U. D'Ortona, W. Kramer, H. J. H. Clercx, K. Schneider and G. J. F. van Heijst, Fourier spectral and wavelet solvers for the incompressible Navier-Stokes equations with volume-penalization: Convergence of a dipole-wall collision. Journal of Computational Physics, Vol. 227, 919-945, 2007.

[48] T. Y. Hou and B. R. Wetton, Stable fourth-order stream-function methods for incompressible flows with boundaries. Journal of Computational Mathematics, Vol. 27, No. 4, 441-458, 2009.

[49] F. Sabetghadam, S. A. Ghaffari and M. Dadashi, Implementation of vortex stretching into the two-dimensional Navier-Stokes equations via arbitrary external straining. Advances in Turbulence XII, Springer Proceedings in Physics, 132, 2009.

[50] M. O. Domingues , O. Roussel and K. Schneider : An adaptive multiresolution method for parabolic PDEs with time-step control. International Journal for Numerical Methods in Engineering, Vol. 78, 652-670, 2009.

[51] K. Brix, S. S. Melian, S. Müller and G. Schieffer, Parallelisation of multiscale-based grid adaptation using space-filling curves. ESAIM: Proceedings, Vol. 29, 108-129, 2009.

[52] K. Schneider and O. V. Vasilyev, Wavelet methods in computational fluid dynamics. Annual Review of Fluid Mechanics, Vol. 42, 473-503, 2010.

[53] B. Hejazialhosseini, D. Rossinelli, M. Bergdorf and P. Koumoutsakos, High order finite volume methods on wavelet-adapted grids with local time-stepping on multicore architectures for the simulation of shock-bubble interactions. Journal of Computational Physics, Vol. 229, 8364-8383, 2010.

[54] M. O. Domingues , S. M. Gomes, O. Roussel and K. Schneider, Adaptive multiresolution methods. ESAIM: Proceedings, Vol. 34, 1-96, 2011.

[55] *The code is developped in FORTRAN and is accessible for all by sending a mail to*: ghaffari@L3m.univ-mrs.fr or s.amin.ghaffary@gmail.com